



Panduan Pengguna

AWS Identity and Access Management



AWS Identity and Access Management: Panduan Pengguna

Copyright © 2024 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 para pelanggan, atau dengan cara apa pun yang menghina atau mendiskreditkan Amazon. Semua merek dagang lain yang tidak dimiliki oleh Amazon merupakan properti dari masing-masing pemilik, yang mungkin berafiliasi, terkait dengan, atau disponsori oleh Amazon, atau tidak.

Table of Contents

Apa itu IAM?	1
Pengantar video untuk IAM	2
Fitur IAM	2
Mengakses IAM	4
Kapan saya menggunakan IAM	5
Saat Anda melakukan fungsi pekerjaan yang berbeda	5
Ketika Anda berwenang untuk mengakses AWS sumber daya	6
Saat Anda masuk sebagai pengguna IAM	6
Ketika Anda mengambil peran IAM	7
Saat Anda membuat kebijakan dan izin	9
Cara kerja IAM	10
Ketentuan	11
Utama	13
Permintaan	13
Autentikasi	14
Otorisasi	14
Tindakan atau operasi	15
Sumber daya	16
Pengguna di AWS	16
Hanya untuk akses pertama kali: Kredensial pengguna akar Anda	16
Pengguna dan pengguna IAM di IAM Identity Center	17
Menggabungkan pengguna yang ada	17
Metode kontrol akses	19
Izin dan kebijakan di IAM	23
Kebijakan dan akun	23
Kebijakan dan pengguna	24
Kebijakan dan grup	24
Pengguna gabungan dan peran	25
Kebijakan berbasis identitas dan berbasis sumber daya	25
Apa itu ABAC?	26
Membandingkan ABAC dengan model RBAC tradisional	27
Fitur keamanan di luar IAM	29
Tautan cepat ke tugas umum	30
Pencarian konsol IAM	33

Menggunakan pencarian konsol IAM	34
Ikon di hasil pencarian konsol IAM	34
Contoh frasa pencarian	35
AWS CloudFormation sumber daya	36
IAM dan template AWS CloudFormation	36
Pelajari lebih lanjut tentang AWS CloudFormation	37
Menggunakan AWS CloudShell	37
Memperoleh izin IAM untuk AWS CloudShell	38
Berinteraksi dengan IAM menggunakan AWS CloudShell	38
Bekerja dengan AWS SDK	40
Menyiapkan	42
Mendaftar Akun AWS	43
Membuat pengguna administratif	43
Bersiaplah untuk izin dengan hak istimewa paling sedikit	44
Metode manajemen IAM	45
Konsol AWS	45
AWSAntarmuka Baris Perintah (CLI) dan Kit Pengembangan Perangkat Lunak (SDK)	47
Akun AWS ID Anda dan aliasnya	49
Lihat Akun AWS ID Anda	50
Tentang alias akun	51
Membuat, menghapus, dan membuat daftar alias Akun AWS	52
Memulai	56
Prasyarat	56
Buat pengguna IAM pertama Anda	56
Buat peran pertama Anda	58
Buat kebijakan IAM pertama Anda	61
Akses terprogram	62
Praktik terbaik keamanan dan kasus penggunaan	64
Praktik terbaik keamanan	64
Mewajibkan pengguna manusia untuk menggunakan federasi dengan penyedia identitas untuk mengakses AWS menggunakan kredensi sementara	65
Memerlukan beban kerja untuk menggunakan kredensial sementara dengan peran IAM untuk mengakses AWS	66
Membutuhkan otentikasi multi-faktor (MFA)	66
Perbarui kunci akses bila diperlukan untuk kasus penggunaan yang memerlukan kredensi jangka panjang	67

Ikuti praktik terbaik untuk melindungi kredensial pengguna root Anda	68
Menerapkan izin hak istimewa paling sedikit	68
Memulai kebijakan AWS terkelola dan beralih ke izin hak istimewa paling sedikit	68
Gunakan IAM Access Analyzer untuk menghasilkan kebijakan hak istimewa paling sedikit berdasarkan aktivitas akses	69
Tinjau dan hapus pengguna, peran, izin, kebijakan, dan kredensial yang tidak digunakan secara teratur	69
Gunakan ketentuan dalam kebijakan IAM untuk membatasi akses lebih lanjut	69
Memverifikasi akses publik dan lintas akun dengan IAM Access Analyzer	70
Gunakan IAM Access Analyzer untuk memvalidasi kebijakan IAM Anda guna memastikan izin yang aman dan fungsional	70
Menetapkan pagar pembatas izin di beberapa akun	70
Gunakan batas izin untuk mendelegasikan manajemen izin dalam akun	71
Praktik terbaik pengguna root	71
Amankan kredensi pengguna root Anda untuk mencegah penggunaan yang tidak sah	72
Gunakan kata sandi pengguna root yang kuat untuk membantu melindungi akses	73
Amankan login pengguna root Anda dengan otentikasi multi-faktor (MFA)	73
Jangan membuat kunci akses untuk pengguna root	74
Gunakan persetujuan multi-orang untuk login pengguna root sedapat mungkin	74
Menggunakan alamat email grup untuk kredensi pengguna root	74
Batasi akses ke mekanisme pemulihan akun	74
Amankan kredensi pengguna root akun Organizations Anda	75
Memantau akses dan penggunaan	76
Kasus penggunaan bisnis	77
Pengaturan awal example corp	78
Gunakan kasus untuk IAM dengan Amazon EC2	79
Gunakan kasus untuk IAM dengan Amazon S3	80
Tutorial	83
Berikan akses ke konsol penagihan	83
Prasyarat	85
Langkah 1: Aktifkan akses IAM ke informasi penagihan di akun pengujian Anda AWS	85
Langkah 2: Buat pengguna dan grup uji	86
Langkah 3: Buat peran untuk memberikan akses ke AWS Billing konsol	88
Langkah 4: Uji akses ke konsol	89
Ringkasan	91
Sumber daya terkait	91

Mendelegasikan akses di Akun AWS menggunakan peran	91
Prasyarat	93
Buat peran dalam Akun Produksi	94
Berikan akses ke peran tersebut	98
Akses uji dengan mengalihkan peran	100
Sumber daya terkait	105
Ringkasan	106
Buat kebijakan terkelola pelanggan	106
Prasyarat	107
Langkah 1: Buat kebijakan	107
Langkah 2: Lampirkan kebijakan	108
Langkah 3: Uji akses pengguna	109
Sumber daya terkait	109
Ringkasan	109
Gunakan kontrol akses berbasis atribut (ABAC)	109
Ikhtisar tutorial	110
Prasyarat	112
Langkah 1: Buat pengguna uji	113
Langkah 2: Buat kebijakan ABAC	115
Langkah 3: Buat peran	119
Langkah 4: Uji pembuatan rahasia	120
Langkah 5: Uji menampilkan rahasia	123
Langkah 6: Uji skalabilitas	125
Langkah 7: Uji memperbarui dan menghapus rahasia	127
Ringkasan	129
Sumber daya terkait	129
Menggunakan tanda sesi SAML untuk ABAC	130
Mengizinkan pengguna mengelola kredensial dan pengaturan MFA mereka	135
Prasyarat	136
Langkah 1: Buat kebijakan untuk menerapkan masuk MFA	136
Langkah 2: Lampirkan kebijakan ke grup pengguna uji Anda	138
Langkah 3: Uji akses pengguna Anda	138
Sumber daya terkait	140
Identitas	141
Akun AWS pengguna root	142
Pengguna IAM	142

Grup pengguna IAM	143
Peran IAM	143
Kredensi sementara di IAM	144
Kapan menggunakan pengguna IAM Identity Center?	145
Kapan harus membuat pengguna IAM (alih-alih peran)	145
Kapan harus membuat sebuah peran IAM (alih-alih pengguna)	146
Bandingkan Pengguna root akun AWS dan pengguna IAM	147
Pengguna root akun AWS	148
Aktifkan perangkat MFA virtual untuk Pengguna root akun AWS (konsol) Anda	149
Aktifkan token TOTP perangkat keras untuk pengguna Akun AWS root (konsol)	152
Aktifkan kunci keamanan FIDO untuk pengguna Akun AWS root (konsol)	154
Ubah kata sandi	156
Mengatur ulang kata sandi pengguna akar yang hilang atau terlupakan	158
Membuat kunci akses untuk pengguna root	160
Menghapus kunci akses untuk pengguna root	162
Tugas yang membutuhkan pengguna root	164
Memecahkan masalah pengguna root	165
Informasi terkait	166
Pengguna	167
Bagaimana AWS mengidentifikasi pengguna IAM	167
Pengguna dan kredensialnya IAM	168
Pengguna dan izin IAM	169
Pengguna dan akun IAM	170
Pengguna IAM sebagai akun layanan	170
Menambahkan pengguna	170
Mengontrol akses pengguna ke konsol	178
Cara pengguna IAM M M M IAM IAM IAM IAM IAMAWS	180
Mengelola pengguna	184
Mengubah izin untuk pengguna	191
Mengelola kata sandi	198
Tombol akses	214
Mengambil kata sandi atau access key yang hilang	232
Autentikasi multi-faktor (MFA)	233
Temukan kredensi yang tidak digunakan	310
Mendapatkan laporan kredensial	314
Menggunakan IAM dengan CodeCommit	320

Menggunakan IAM dengan Amazon Keyspaces	324
Mengelola sertifikat server	325
Grup pengguna	332
Membuat grup pengguna	334
Mengelola grup pengguna	336
Peran	343
Istilah dan konsep	345
Skenario umum	349
Peran terkait layanan	369
Membuat peran	382
Menggunakan peran	423
Mengelola peran	595
Penyedia dan federasi identitas	619
Federasi dengan Pusat Identitas IAM	621
Federasi dengan IAM	621
Federasi dengan kumpulan identitas Amazon Cognito	622
Skenario umum	622
Federasi OIDC	628
Federasi SAML 2.0	646
Kredensial keamanan sementara	678
AWS STS dan AWS daerah	679
Skenario umum untuk kredensial sementara	679
Meminta kredensial keamanan sementara	682
Menggunakan kredensial sementara dengan sumber daya AWS	699
Mengontrol izin untuk kredensial keamanan sementara	704
Mengelola AWS STS dalam Wilayah AWS	737
Menggunakan token pembawa	746
Contoh aplikasi yang menggunakan kredensial sementara	747
Mengaktifkan akses broker identitas khusus ke konsol AWS	748
Sumber daya tambahan untuk kredensial sementara	763
Menandai sumber daya IAM	764
Pilih konvensi penamaan tanda AWS	765
Aturan untuk menandai di IAM dan AWS STS	766
Menandai pengguna IAM	769
Menandai peran IAM	772
Menandai kebijakan yang dikelola pelanggan	776

Menandai penyedia identitas IAM	779
Menandai profil instans	785
Menandai sertifikat server	788
Menandai perangkat MFA virtual	790
Tanda sesi	793
Log peristiwa dengan CloudTrail	807
IAM dan AWS STS informasi di CloudTrail	807
Pencatatan permintaan IAM dan AWS STS API	808
Mencatat permintaan API ke layanan AWS lainnya	809
Mencatat peristiwa masuk pengguna	809
Mencatat peristiwa masuk untuk kredensial sementara	810
Contoh peristiwa API IAM dalam catatan CloudTrail	812
Contoh peristiwa AWS STS API di CloudTrail log	813
Contoh peristiwa masuk dalam catatan CloudTrail	823
Perilaku kebijakan kepercayaan peran IAM	825
Manajemen akses	827
Sumber daya manajemen akses	828
Kebijakan dan Izin	829
Jenis kebijakan	829
Kebijakan dan pengguna root	835
Gambaran dari kebijakan JSON	835
Berikan hak istimewa paling rendah	840
Kebijakan terkelola dan kebijakan inline	842
Batas izin	852
Identitas vs sumber daya	866
Mengontrol akses menggunakan kebijakan	870
Kontrol akses ke pengguna dan peran IAM menggunakan tag	882
Kontrol akses ke AWS sumber daya menggunakan tag	885
Akses sumber daya lintas akun	890
Teruskan sesi akses	897
Contoh kebijakan	900
Mengelola kebijakan IAM	980
Membuat kebijakan IAM	981
Memvalidasi kebijakan	991
Membuat kebijakan	992
Menguji Kebijakan IAM	993

Menambah atau menghapus izin identitas	1009
Peningkatan versi IAM	1021
Menyunting Kebijakan IAM	1025
Menghapus kebijakan IAM	1032
Menyaring izin menggunakan informasi akses	1036
Memahami kebijakan	1576
Ringkasan kebijakan (daftar layanan)	1577
Ringkasan layanan (daftar tindakan)	1591
Ringkasan tindakan (daftar sumber daya)	1597
Contoh ringkasan kebijakan	1600
Izin diperlukan	1610
Izin untuk mengurus identitas IAM	1610
Izin untuk bekerja di AWS Management Console	1612
Memberikan izin di seluruh akun AWS	1613
Izin satu layanan untuk mengakses layanan lainnya	1613
Tindakan yang diperlukan	1614
Contoh kebijakan untuk IAM	1615
Contoh kode	1620
IAM	1625
Tindakan	1640
Skenario	2191
AWS STS	2544
Tindakan	2545
Skenario	2573
Keamanan	2591
Kredensial keamanan AWS	2592
Pertimbangan keamanan	2593
Identitas federasi	2594
Autentikasi multi-faktor (MFA)	2594
Akses terprogram	2595
Alternatif untuk kunci akses jangka panjang	2597
Mengakses AWS menggunakan kredensi Anda AWS	2598
AWS pedoman audit keamanan	2599
Kapan melakukan audit keamanan	2600
Pedoman untuk audit	2600
Tinjau kredensi AWS akun Anda	2600

Tinjau pengguna IAM Anda	2601
Tinjau grup IAM Anda	2602
Tinjau IAM role Anda	2602
Tinjau penyedia IAM untuk SAML and OpenID Connect (OIDC) Anda	2602
Tinjau aplikasi seluler Anda	2602
Tips untuk meninjau kebijakan IAM	2603
Perlindungan data	2605
Enkripsi data dalam IAM dan AWS STS	2606
Manajemen kunci dalam IAM dan AWS STS	2606
Privasi lalu lintas internetwork di IAM dan AWS STS	2606
Pencatatan dan pemantauan	2607
Validasi kepatuhan	2608
Ketahanan	2609
Praktik terbaik untuk ketahanan IAM	2611
Keamanan infrastruktur	2611
Analisis konfigurasi dan kerentanan	2612
AWS kebijakan terkelola	2613
IAM ReadOnlyAccess	2613
IAM UserChangePassword	2614
IAM AccessAnalyzerFullAccess	2614
IAM AccessAnalyzerReadOnlyAccess	2616
AccessAnalyzerServiceRolePolicy	2617
.....	2620
Pembaruan kebijakan	2620
IAM Access Analyzer	2624
Mengidentifikasi sumber daya yang dibagikan dengan entitas eksternal	2624
Mengidentifikasi akses yang tidak terpakai yang diberikan kepada pengguna dan peran IAM ..	2627
Memvalidasi kebijakan terhadap praktik AWS terbaik	2627
Memvalidasi kebijakan terhadap standar keamanan yang Anda tentukan	2628
Membuat kebijakan	2628
Harga untuk IAM Access Analyzer	2628
Temuan untuk akses eksternal dan tidak terpakai	2629
Cara kerja temuan IAM Access Analyzer	2631
Memulai dengan temuan IAM Access Analyzer	2632
Dasbor temuan	2639
Bekerja dengan temuan	2643

Meninjau temuan	2644
Memfilter temuan	2648
Mengarsipkan temuan	2653
Menyelesaikan temuan	2653
Jenis sumber daya yang mendukung	2655
Pengaturan	2663
Aturan arsip	2665
Pemantauan EventBridge dengan	2668
Integrasi Security Hub	2677
Logging dengan CloudTrail	2685
Kunci filter IAM Access Analyzer	2688
Menggunakan peran terkait layanan	2697
Akses pratinjau	2699
Pratinjau akses di konsol Amazon S3	2700
Akses pratinjau dengan IAM Access Analyzer API	2701
Cek untuk memvalidasi kebijakan	2705
Validasi kebijakan IAM Access Analyzer	2706
Pemeriksaan kebijakan khusus	2812
Pembuatan kebijakan IAM Access Analyzer	2815
Cara kerja pembuatan kebijakan	2816
Informasi tingkat tindakan dan layanan	2817
Hal yang perlu diketahui	2817
Izin diperlukan	2818
Menghasilkan kebijakan berdasarkan CloudTrail aktivitas (konsol)	2821
Buat kebijakan menggunakan AWS CloudTrail data di akun lain	2825
Menghasilkan kebijakan berdasarkan CloudTrail aktivitas (AWS CLI)	2829
Menghasilkan kebijakan berdasarkan CloudTrail aktivitas (AWS API)	2829
Layanan pembuatan kebijakan IAM Access Analyzer	2830
Kuota Penganalisis Akses IAM	2841
Menyelesaikan masalah IAM	2843
Masalah umum	2843
Saya tidak dapat masuk ke akun AWS saya	2844
Saya kehilangan access key saya	2844
Variabel kebijakan tidak berfungsi	2844
Perubahan yang saya buat tidak selalu langsung terlihat	2845
Saya tidak berwenang untuk melakukan: iam:DeleteVirtualMFADevice	2846

Bagaimana cara membuat pengguna IAM dengan aman?	2846
Sumber daya tambahan	2847
Akses pesan kesalahan ditolak	2848
Saya mendapatkan “akses ditolak” ketika saya mengajukan permintaan ke AWS layanan .	2848
Saya mendapatkan “akses ditolak” ketika saya membuat permintaan dengan kredensial keamanan sementara	2850
Akses ditolak contoh	2851
Kebijakan IAM	2857
Atasi masalah menggunakan editor visual	2858
Pemecahan masalah menggunakan ringkasan kebijakan	2864
Pemecahan masalah manajemen kebijakan	2873
Memecahkan masalah pada dokumen kebijakan JSON	2874
Kunci keamanan FIDO	2880
Saya tidak dapat mengaktifkan kunci keamanan FIDO saya	2880
Saya tidak dapat masuk menggunakan kunci keamanan FIDO saya	2881
Saya menghilangkan atau merusak kunci keamanan FIDO saya	2882
Masalah lainnya	2882
Peran IAM	2882
Saya tidak dapat mengsumsikan peran	2883
Peran baru muncul di akun AWS saya	2885
Saya tidak dapat mengedit atau menghapus peran di Akun AWS	2885
Saya tidak berwenang untuk melakukan: iam: PassRole	2886
Mengapa saya tidak dapat mengasumsikan peran dengan sesi 12 jam? (API AWS CLI, AWS)	2887
Saya menerima kesalahan saat mencoba beralih peran di konsol IAM	2887
Peran saya memiliki kebijakan yang memungkinkan saya melakukan tindakan, tetapi saya mendapatkan “akses ditolak”	2888
Layanan tidak membuat versi kebijakan default peran tersebut	2888
Tidak ada kasus penggunaan untuk peran layanan di konsol	2890
IAM dan Amazon EC2	2891
Saat mencoba meluncurkan instans, saya tidak melihat peran yang diharapkan di daftar Peran IAM konsol Amazon EC2	2891
Kredensial pada instans saya untuk peran yang salah	2892
Saat saya mencoba memanggil <code>AddRoleToInstanceProfile</code> , saya mendapatkan kesalahan <code>AccessDenied</code>	2892

Amazon EC2: Ketika saya mencoba meluncurkan instans dengan peran, saya mendapatkan kesalahan <code>AccessDenied</code>	2893
Saya tidak dapat mengakses kredensial keamanan sementara di instans EC2 saya	2893
Apa arti kesalahan dari dokumen <code>info</code> di subtree IAM?	2894
IAM dan Amazon S3	2895
Bagaimana saya memberikan akses anonim ke bucket Amazon S3?	2896
Saya masuk sebagai pengguna Akun AWS utama; mengapa saya tidak dapat mengakses bucket Amazon S3 dalam akun saya?	2896
Federasi SAML 2.0	2896
Respons SAML tidak valid	2897
RoleSessionName diperlukan	2898
Tidak diizinkan untuk AssumeRoleWith SAMP	2898
Karakter tidak valid RoleSessionName	2899
Karakter Identitas Sumber Tidak Valid	2899
Tanda tangan respons tidak valid	2900
Gagal mengambil peran	2900
Tidak dapat mengurai metadata	2900
Penyedia yang ditentukan tidak ada	2901
DurationSeconds melebihi MaxSessionDuration	2901
Respons tidak mengandung audiens yang diperlukan	2901
Melihat respons SAML di peramban Anda	2901
Referensi	2905
Amazon Resource Name (ARN)	2905
Format ARN	2905
Cari format ARN untuk sumber daya	2907
Jalur di ARN	2907
Pengidentifikasi IAM	2908
Nama dan jalur yang ramah	2908
ARN IAM	2909
Pengidentifikasi unik	2915
IAM dan AWS STS kuota	2919
Persyaratan nama IAM	2919
Kuota objek IAM	2920
Kuota Penganalisis Akses IAM	2921
Kuota Peran IAM Di Mana Saja	2921
Batas karakter IAM dan STS	2922

Titik akhir VPC antarmuka	2927
Ketersediaan	2928
Buat titik akhir VPC untuk AWS STS	2929
Layanan yang bekerja dengan IAM	2929
Layanan yang bekerja dengan IAM	2931
Informasi lain	2982
Menandatangani permintaan AWS API	2986
Kapan menandatangani permintaan	2988
Mengapa permintaan ditandatangani	2988
Elemen permintaan Signature Version 4	2989
Metode otentikasi	2991
Buat permintaan yang ditandatangani	2996
Minta contoh tanda tangan	3007
Pemecahan Masalah	3009
Referensi kebijakan	3014
Referensi elemen JSON	3015
Logika evaluasi kebijakan	3088
Tata bahasa kebijakan	3112
AWS kebijakan terkelola untuk fungsi pekerjaan	3121
Kunci kepatuhan global	3138
Kunci kondisi IAM	3201
Tindakan, sumber daya, dan kunci kondisi	3227
Sumber daya	3228
Identitas	3228
Kredensial (RDS, access key, dan perangkat MFA)	3228
Izin dan kebijakan	3229
Federasi dan delegasi	3229
IAM dan produk lainnya AWS	3230
Menggunakan IAM dengan Amazon EC2	3230
Menggunakan IAM dengan Amazon S3	3230
Menggunakan IAM dengan Amazon RDS	3230
Menggunakan IAM dengan Amazon DynamoDB	3231
Praktik kerahasiaan umum	3231
Sumber daya umum	3231
Membuat permintaan kueri HTTP	3233
Titik akhir	3234

HTTPS diperlukan	3234
Menandatangani permintaan API IAM	3234
Riwayat dokumen	3236
.....	mmmcclx

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 secara terpusat yang mengontrol AWS sumber daya mana yang dapat diakses pengguna. Anda menggunakan IAM untuk mengontrol siapa yang diautentikasi (masuk) dan diotorisasi (memiliki izin) untuk menggunakan sumber daya.

Saat Anda 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 dan diakses dengan masuk dengan alamat email dan kata sandi yang Anda gunakan untuk membuat akun. Kami sangat menyarankan agar Anda tidak menggunakan pengguna root untuk tugas sehari-hari Anda. 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, lihat [Tugas yang memerlukan kredensial pengguna root](#).

Daftar Isi

- [Pengantar video untuk IAM](#)
- [Fitur IAM](#)
- [Mengakses IAM](#)
- [Kapan saya menggunakan IAM?](#)
- [Cara kerja IAM](#)
- [Gambaran umum manajemen identitas AWS: Pengguna](#)
- [Gambaran umum manajemen akses: Izin dan kebijakan](#)
- [Untuk apa ABAC? AWS](#)
- [Fitur keamanan di luar IAM](#)
- [Tautan cepat ke tugas umum](#)
- [Pencarian konsol IAM](#)
- [Menciptakan AWS Identity and Access Management sumber daya dengan AWS CloudFormation](#)
- [Menggunakan AWS CloudShell untuk bekerja dengan AWS Identity and Access Management](#)
- [Menggunakan IAM dengan SDK AWS](#)

Pengantar video untuk IAM

AWS Pelatihan dan Sertifikasi menyediakan pengenalan video 10 menit untuk IAM:

[Pengantar AWS Identity and Access Management](#)

Fitur IAM

IAM memberikan fitur-fitur berikut untuk Anda:

Akses bersama ke 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 menyelesaikan akses 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 hanya-baca ke beberapa bucket S3, atau izin untuk menjalankan hanya beberapa instans EC2, atau untuk mengakses informasi penagihan Anda tetapi tidak ada yang lainnya.

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 sumber daya lain. AWS Contohnya termasuk bucket S3 dan tabel DynamoDB.

Multi-factor authentication (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 menggunakannya WebAuthn untuk keamanan MFA. Untuk informasi selengkapnya, lihat [Konfigurasi yang didukung untuk menggunakan kunci keamanan FIDO](#).

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 mendapatkan akses sementara ke situs Anda. Akun AWS

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 PCI AWS Compliance Package, lihat [PCI DSS Level 1](#).

Terintegrasi dengan banyak AWS layanan

Untuk daftar AWS layanan yang bekerja dengan IAM, lihat [AWS layanan yang bekerja dengan IAM](#).

Konsisten Pada Akhirnya

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](#).

Gratis untuk digunakan

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

kredensi keamanan AWS STS sementara. Untuk informasi tentang harga AWS produk lain, lihat [halaman harga Amazon Web Services](#).

Mengakses IAM

Anda dapat bekerja AWS Identity and Access Management dengan salah satu cara berikut.

AWS Management Console

Konsol adalah antarmuka berbasis browser untuk mengelola IAM dan sumber daya. AWS Untuk informasi selengkapnya tentang mengakses IAM melalui konsol, lihat [Cara masuk di AWS Sign-In Panduan Pengguna](#). AWS

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 AWS Tools for Windows 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 kredensial 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 Lihat informasi yang lebih lengkap di [Menggunakan AWS CloudShell untuk bekerja dengan AWS Identity and Access Management](#)

AWS SDK

AWS menyediakan SDK (perangkat pengembangan perangkat lunak) yang terdiri dari pustaka dan kode sampel untuk berbagai bahasa dan platform pemrograman (Java, Python, 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, 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](#).

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 kredensial Anda. Untuk informasi lebih lanjut, lihat [Memanggil API IAM menggunakan permintaan kueri HTTP](#) dan [Referensi IAM API](#).

Kapan saya menggunakan IAM?

Saat 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 kredensial 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 AWS sumber daya di perusahaan Anda, Anda mungkin memiliki akses penuh ke IAM. Tugas Anda adalah menentukan fitur dan sumber daya IAM mana yang harus diakses pengguna layanan Anda. Kemudian, Anda harus mengirimkan permintaan kepada administrator IAM Anda 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 berwenang untuk mengakses AWS sumber daya

Autentikasi merupakan cara Anda untuk masuk ke AWS dengan menggunakan kredensial identitas Anda. Anda harus terautentikasi (masuk ke AWS) sebagai Pengguna root akun AWS, sebagai pengguna IAM, atau dengan mengambil peran IAM.

Anda dapat masuk ke AWS sebagai identitas terfederasi dengan menggunakan kredensial yang disediakan melalui sumber identitas. AWS IAM Identity Center Para pengguna (Pusat Identitas IAM), otentikasi sign-on tunggal perusahaan Anda, dan kredensial Google atau Facebook Anda merupakan contoh identitas terfederasi. Saat Anda masuk sebagai identitas terfederasi, administrator Anda sebelumnya menyiapkan federasi identitas dengan menggunakan peran IAM. Ketika Anda mengakses AWS dengan menggunakan federasi, Anda secara tidak langsung mengambil suatu peran.

Tergantung pada jenis pengguna Anda, Anda dapat masuk ke AWS Management Console atau portal akses AWS. Untuk informasi selengkapnya tentang masuk ke AWS, silakan lihat [Cara masuk ke Akun AWS Anda](#) di Panduan Pengguna AWS Sign-In.

Jika Anda mengakses AWS secara terprogram, AWS sediakan kit pengembangan perangkat lunak (SDK) dan antarmuka baris perintah (CLI) untuk menandatangani permintaan Anda secara kriptografis dengan menggunakan kredensial Anda. Jika Anda tidak menggunakan peralatan AWS, maka Anda harus menandatangani sendiri permintaan tersebut. Untuk informasi selengkapnya tentang penggunaan metode yang disarankan untuk menandatangani permintaan sendiri, silakan lihat [Menandatangani permintaan API AWS](#) di Panduan Pengguna IAM.

Terlepas dari metode autentikasi yang Anda gunakan, Anda mungkin juga diminta untuk menyediakan informasi keamanan tambahan. Sebagai contoh, AWS menyarankan supaya Anda menggunakan autentikasi multi-faktor (MFA) untuk meningkatkan keamanan akun Anda. Untuk mempelajari selengkapnya, silakan lihat [Autentikasi multi-faktor](#) di Panduan Pengguna AWS IAM Identity Center dan [Menggunakan autentikasi multi-faktor \(MFA\) di AWS](#) di Panduan Pengguna IAM.

Saat Anda masuk sebagai pengguna IAM

[Pengguna IAM](#) adalah identitas dalam Akun AWS Anda yang memiliki izin khusus untuk satu orang atau aplikasi. Apabila memungkinkan, kami menyarankan untuk mengandalkan pada kredensial temporer alih-alih membuat pengguna IAM yang memiliki kredensial jangka panjang seperti kata sandi dan kunci akses. Namun, jika Anda memiliki kasus penggunaan khusus yang memerlukan kredensial jangka panjang dengan pengguna IAM, kami menyarankan Anda memutar kunci akses.

Untuk informasi selengkapnya, silakan lihat [Memutar kunci akses secara teratur untuk kasus penggunaan yang memerlukan kredensial jangka panjang](#) di Panduan Pengguna IAM.

[Grup IAM](#) adalah identitas yang menerangkan secara spesifik kumpulan pengguna IAM. Anda tidak dapat masuk sebagai kelompok. Anda dapat menggunakan grup untuk menerangkan secara spesifik izin untuk beberapa pengguna sekaligus. Grup membuat izin lebih mudah dikelola untuk sekelompok besar pengguna. Sebagai contoh, Anda dapat memiliki grup yang diberi nama AdminIAM dan memberikan izin kepada grup tersebut untuk mengelola sumber daya IAM.

Pengguna berbeda dari peran. Pengguna secara unik terkait dengan satu orang atau aplikasi, tetapi peran tersebut dimaksudkan untuk dapat digunakan oleh siapa pun yang membutuhkannya. Pengguna memiliki kredensial jangka panjang permanen, tetapi peran memberikan kredensial temporer. Untuk mempelajari selengkapnya, silakan lihat [Kapan harus membuat pengguna IAM \(alih-alih peran\)](#) di Panduan Pengguna IAM.

Ketika Anda mengambil peran IAM

[Peran IAM](#) merupakan identitas dalam Akun AWS Anda yang memiliki izin khusus. Peran ini mirip dengan pengguna IAM, tetapi tidak terkait dengan orang tertentu. Anda dapat menggunakan peran IAM untuk sementara dalam AWS Management Console dengan [berganti peran](#). Anda dapat mengambil peran dengan cara memanggil operasi API AWS CLI atau AWS atau menggunakan URL kustom. Untuk informasi selengkapnya tentang cara menggunakan peran, silakan lihat [menggunakan peran IAM](#) di Panduan Pengguna IAM.

IAM role dengan kredensial temporer berguna dalam situasi berikut:

- Akses pengguna gabungan – Untuk menetapkan izin ke sebuah identitas terfederasi, Anda harus membuat sebuah peran dan menentukan izin untuk peran tersebut. Ketika identitas gabungan terfederasi mengautentikasi, identitas tersebut terhubung dengan peran dan diberikan izin yang ditentukan oleh peran. Untuk informasi tentang peran-peran untuk federasi, silakan lihat [Membuat sebuah peran untuk Penyedia Identitas pihak ketiga](#) di Panduan Pengguna IAM. Jika Anda menggunakan Pusat Identitas IAM, Anda mengonfigurasi serangkaian izin. Untuk mengontrol apa yang dapat diakses identitas Anda setelah identitas tersebut diautentikasi, Pusat Identitas IAM mengkorelasikan izin yang diatur ke peran dalam IAM. Untuk informasi tentang rangkaian izin, silakan lihat [Rangkaian izin](#) di Panduan Pengguna AWS IAM Identity Center.
- Izin pengguna IAM sementara – Pengguna atau peran IAM dapat mengambil peran IAM untuk sementara mengambil izin berbeda untuk tugas tertentu.

- Akses lintas akun – Anda dapat menggunakan peran IAM untuk mengizinkan seseorang (pengguna utama tepercaya) di akun berbeda untuk mengakses sumber daya yang ada di akun Anda. Peran adalah cara utama untuk memberikan akses lintas akun. Namun, pada beberapa Layanan AWS, Anda dapat melampirkan kebijakan secara langsung ke sumber daya (alih-alih menggunakan suatu peran sebagai proksi). Untuk mempelajari perbedaan antara kebijakan peran dan kebijakan berbasis sumber daya untuk akses lintas akun, silakan lihat [Bagaimana peran IAM role berbeda dari kebijakan berbasis sumber daya](#) dalam Panduan Pengguna IAM.
- Akses lintas layanan – Sebagian Layanan AWS menggunakan fitur di Layanan AWS lainnya. Sebagai contoh, ketika Anda melakukan panggilan dalam suatu layanan, lazim pada layanan tersebut menjalankan aplikasi di Amazon EC2 atau menyimpan objek di Amazon S3. Suatu layanan mungkin melakukan hal tersebut menggunakan izin pengguna utama panggilan, menggunakan peran layanan, atau peran tertaut layanan.
- Sesi akses maju (FAS) – Ketika Anda menggunakan pengguna IAM atau peran IAM untuk melakukan tindakan-tindakan di AWS, Anda akan dianggap sebagai seorang pengguna utama. Ketika Anda menggunakan beberapa layanan, Anda mungkin melakukan sebuah tindakan yang kemudian dilanjutkan oleh tindakan lain pada layanan yang berbeda. FAS menggunakan izin dari pengguna utama untuk memanggil Layanan AWS, yang dikombinasikan dengan Layanan AWS yang diminta untuk membuat pengajuan ke layanan hilir. Permintaan FAS hanya diajukan ketika sebuah layanan menerima pengajuan yang memerlukan interaksi dengan Layanan AWS lain atau sumber daya lain untuk diselesaikan. Dalam hal ini, Anda harus memiliki izin untuk melakukan kedua tindakan tersebut. Untuk detail kebijakan ketika mengajukan permintaan FAS, silakan lihat [Meneruskan sesi akses](#).
- Peran layanan – Sebuah peran layanan adalah sebuah [peran IAM](#) yang dijalankan oleh suatu layanan untuk melakukan tindakan atas nama Anda. Administrator IAM dapat membuat, memodifikasi, dan menghapus peran layanan dari dalam IAM. Untuk informasi selengkapnya, silakan lihat [Membuat sebuah peran untuk mendelegasikan izin ke Layanan AWS](#) dalam Panduan pengguna IAM.
- Peran tertaut layanan – Peran tertaut layanan adalah tipe peran layanan yang tertaut dengan Layanan AWS. Layanan tersebut dapat menjalankan peran untuk melakukan sebuah tindakan atas nama Anda. Peran tertaut layanan akan muncul di Akun AWS Anda dan dimiliki oleh layanan tersebut. Administrator IAM dapat melihat, tetapi tidak dapat mengedit izin untuk peran tertaut layanan.
- Aplikasi yang berjalan di Amazon EC2 – Anda dapat menggunakan peran IAM untuk mengelola kredensial temporer untuk aplikasi yang berjalan di instans EC2 dan mengajukan permintaan AWS CLI atau API AWS. Cara ini lebih baik daripada menyimpan kunci akses dalam instans

EC2. Untuk menugaskan sebuah peran AWS ke instans EC2 dan membuatnya tersedia untuk semua aplikasinya, Anda dapat membuat sebuah profil instans yang dilampirkan ke instans. Profil instans berisi peran dan memungkinkan program yang berjalan di instans EC2 untuk mendapatkan kredensial sementara. Untuk informasi selengkapnya, silakan lihat [Menggunakan peran IAM untuk memberikan izin ke aplikasi yang berjalan di instans Amazon EC2](#) di Panduan Pengguna IAM.

Untuk mempelajari apakah kita harus menggunakan peran IAM atau pengguna IAM, silakan lihat [Kapan harus membuat peran IAM \(alih-alih pengguna\)](#) di Panduan Pengguna IAM.

Saat Anda membuat kebijakan dan izin

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

Kebijakan ini digunakan dengan peran IAM:

- Kebijakan kepercayaan — Mendefinisikan [prinsip](#) mana yang dapat mengambil peran, dan dalam kondisi apa. Kebijakan kepercayaan adalah jenis kebijakan berbasis sumber daya khusus untuk peran IAM. Peran hanya dapat memiliki satu kebijakan kepercayaan.
- Kebijakan berbasis identitas (sebaris dan terkelola) — Kebijakan ini menentukan izin yang dapat dilakukan oleh 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 diperlukan, pilih lihat kebijakan untuk melihat JSON untuk kebijakan tersebut. Anda dapat menggunakan dokumen kebijakan JSON sebagai contoh untuk kebijakan Anda sendiri.

Note

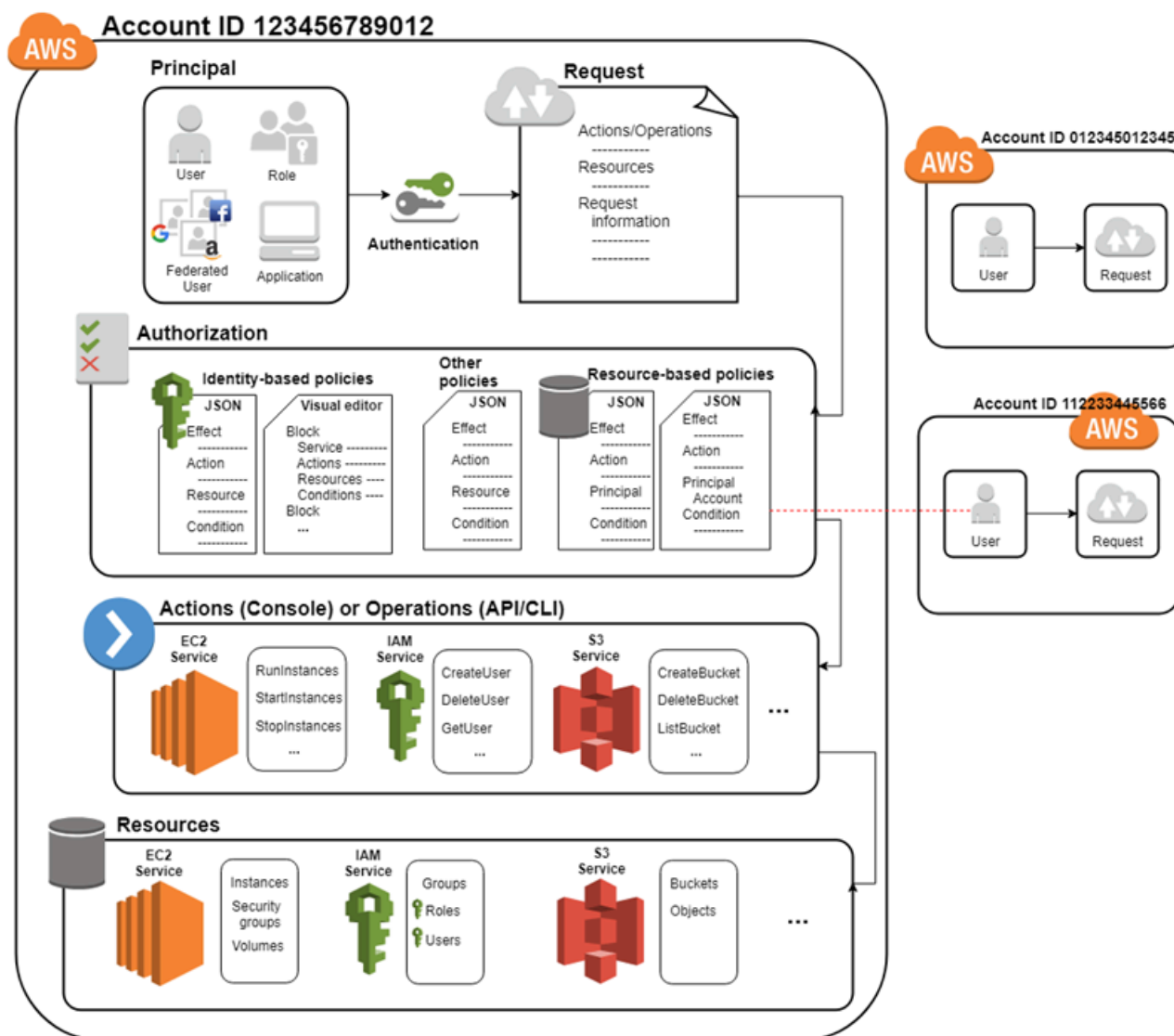
Jika Anda menggunakan Pusat Identitas IAM untuk mengelola pengguna, Anda menetapkan set izin di Pusat Identitas IAM alih-alih melampirkan kebijakan izin ke prinsipal. Saat Anda menetapkan izin yang disetel ke grup atau pengguna di Pusat Identitas AWS IAM, Pusat Identitas IAM akan membuat peran IAM yang sesuai di setiap akun, dan melampirkan kebijakan yang ditentukan dalam izin yang disetel ke peran tersebut. IAM Identity Center

mengelola peran, dan memungkinkan pengguna resmi yang telah Anda tentukan untuk mengambil peran tersebut. Jika Anda mengubah set izin, Pusat Identitas IAM memastikan bahwa kebijakan dan peran IAM terkait diperbarui sesuai dengan itu.

Untuk informasi selengkapnya tentang Pusat Identitas IAM, lihat [Apa itu Pusat Identitas IAM?](#) dalam AWS IAM Identity Center User Guide.

Cara kerja IAM

IAM menyediakan infrastruktur yang diperlukan untuk mengontrol autentikasi dan otorisasi untuk Anda. Akun AWS Infrastruktur IAM diilustrasikan dengan diagram berikut:



Pertama, pengguna manusia atau aplikasi menggunakan kredensi masuk mereka untuk mengotentikasi. AWS Otentikasi disediakan dengan mencocokkan kredensi masuk ke prinsipal (pengguna IAM, pengguna federasi, peran IAM, atau aplikasi) yang dipercaya oleh Akun AWS

Selanjutnya, permintaan dibuat untuk memberikan akses utama ke sumber daya. Akses diberikan sebagai tanggapan atas permintaan otorisasi. 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 terlihat untuk melihat apakah identitas Anda ada dalam daftar pengguna yang berwenang, 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 Anda Akun AWS atau dari orang lain Akun AWS yang Anda percayai.

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

Konsep Basic

- [Ketentuan](#)
- [Utama](#)
- [Permintaan](#)
- [Autentikasi](#)
- [Otorisasi](#)
- [Tindakan atau operasi](#)
- [Sumber daya](#)

Ketentuan

Istilah IAM ini biasanya digunakan saat bekerja dengan AWS:

Sumber Daya IAM

Sumber daya IAM disimpan dalam IAM. Anda dapat menambahkan, mengedit, dan menghapusnya dari IAM.

- pengguna

- grup
- peran
- kebijakan
- objek penyedia identitas

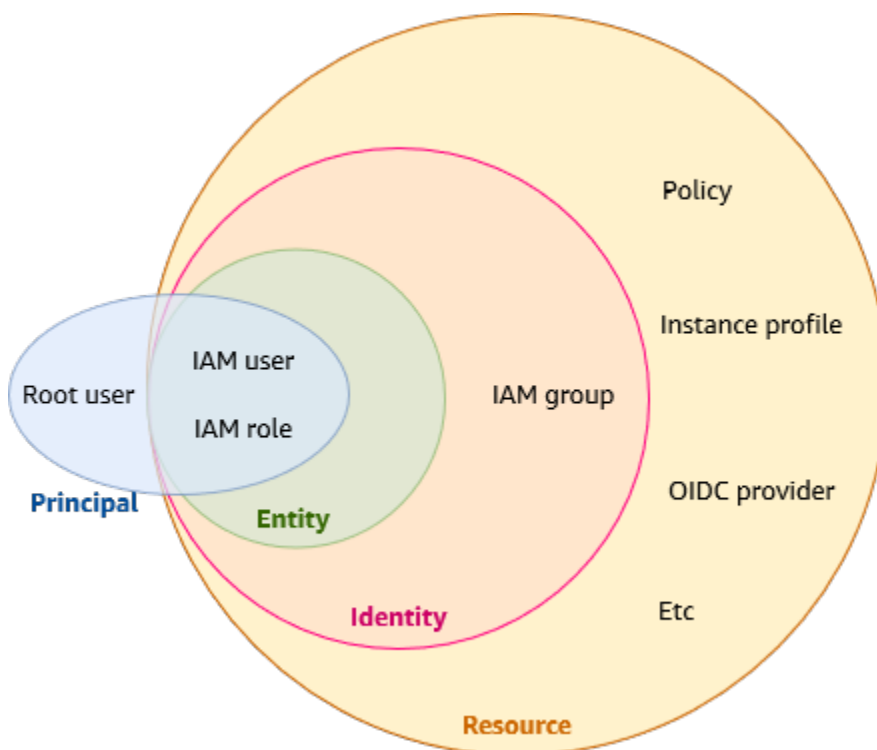
Entitas IAM

Sumber daya IAM yang AWS digunakan untuk otentikasi. Entitas dapat ditentukan sebagai Prinsipal dalam kebijakan berbasis sumber daya.

- pengguna
- peran

Identitas IAM

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



Penanggung jawab

Seseorang atau aplikasi yang menggunakan Pengguna root akun AWS, pengguna IAM, atau peran IAM untuk masuk dan mengajukan permintaan. AWS Penanggung jawab mencakup pengguna gabungan dan peran yang diambil.

Pengguna manusia

Juga dikenal sebagai identitas manusia; orang, administrator, pengembang, operator, dan konsumen aplikasi Anda.

Beban kerja

Kumpulan sumber daya dan kode yang memberikan nilai bisnis, seperti aplikasi atau proses backend. Dapat mencakup aplikasi, alat operasional, dan komponen.

Utama

Prinsipal adalah pengguna manusia atau beban kerja yang dapat membuat permintaan untuk tindakan atau operasi pada AWS sumber daya. Setelah otentikasi, prinsipal dapat diberikan kredensi permanen atau sementara untuk mengajukan permintaan AWS, tergantung pada jenis prinsipalnya. Pengguna IAM dan pengguna root diberikan kredensial permanen, sementara peran diberikan kredensial sementara. Sebagai [praktik terbaik](#), kami menyarankan Anda meminta pengguna manusia dan beban kerja untuk mengakses AWS sumber daya menggunakan kredensial sementara.

Permintaan

Ketika penanggung jawab mencoba menggunakan AWS Management Console, API AWS, atau AWS CLI, dengan permintaan yang dikirim penanggung jawab ke AWS. Permintaan tersebut mencakup informasi berikut:

- Tindakan atau operasi – Tindakan atau operasi yang ingin dilakukan oleh penanggung jawab. Ini bisa menjadi sebuah tindakan di AWS Management Console, atau operasi di AWS CLI atau API AWS.
- Sumber daya — Objek AWS sumber daya di mana tindakan atau operasi dilakukan.
- Penanggung jawab – Orang atau aplikasi yang menggunakan entitas (pengguna atau peran) untuk mengirim permintaan. Informasi tentang penanggung jawab mencakup kebijakan yang berkaitan dengan entitas yang digunakan oleh penanggung jawab untuk masuk.
- Data lingkungan – Informasi tentang alamat IP, agen pengguna, status yang diaktifkan SSL, atau waktu dalam sehari.
- Data sumber daya – Data terkait sumber daya yang diminta. Ini dapat mencakup informasi seperti DynamoDB nama tabel atau tag di instans Amazon EC2.

AWS mengumpulkan informasi permintaan ke konteks permintaan, yang digunakan untuk mengevaluasi dan mengotorisasi permintaan.

Autentikasi

Penanggung jawab harus diautentikasi (masuk ke AWS) menggunakan kredensial mereka untuk mengirim permintaan ke AWS. Beberapa layanan, seperti Amazon S3 dan AWS STS, memungkinkan beberapa permintaan dari pengguna anonim. Namun, mereka adalah pengecualian dari aturan tersebut.

Untuk mengautentikasi dari konsol sebagai pengguna utama, Anda harus masuk dengan alamat email dan kata sandi Anda. Sebagai pengguna federasi, Anda diautentikasi oleh penyedia identitas Anda dan diberikan akses ke AWS sumber daya dengan mengasumsikan peran IAM. Sebagai pengguna IAM, berikan ID akun atau alias Anda, dan kemudian nama pengguna dan kata sandi Anda. Untuk mengautentikasi beban kerja dari API atau AWS CLI, Anda dapat menggunakan kredensial sementara melalui peran yang ditetapkan atau Anda dapat menggunakan kredensial jangka panjang dengan menyediakan kunci akses dan kunci rahasia Anda. Anda juga mungkin diminta untuk memberikan informasi keamanan tambahan. Misalnya, AWS menyarankan agar Anda menggunakan autentikasi Autentikasi Multi-Faktor (MFA) dan kredensial sementara untuk meningkatkan keamanan akun Anda. Untuk mempelajari lebih lanjut tentang entitas IAM yang AWS dapat mengautentikasi, lihat [Pengguna IAM](#) dan [Peran IAM](#)

Otorisasi

Anda juga harus diotorisasi (diizinkan) untuk menyelesaikan permintaan Anda. Selama otorisasi, AWS menggunakan nilai dari konteks permintaan untuk memeriksa kebijakan yang berlaku bagi permintaan. Kemudian menggunakan kebijakan untuk menentukan apakah akan mengizinkan atau menolak permintaan. Sebagian besar kebijakan disimpan di AWS sebagai [dokumen JSON](#) dan menentukan izin untuk entitas penanggung jawab. Ada [beberapa jenis kebijakan](#) yang dapat memengaruhi apakah permintaan diotorisasi. Untuk memberikan izin kepada pengguna Anda untuk mengakses sumber daya AWS di akun mereka sendiri, Anda hanya memerlukan kebijakan berbasis identitas. Kebijakan berbasis sumber daya populer untuk menghibahkan [akses lintas akun](#). Jenis kebijakan lainnya adalah fitur lanjutan dan harus digunakan dengan hati-hati.

AWS memeriksa setiap kebijakan yang berlaku pada konteks permintaan Anda. Jika satu kebijakan izin mencakup tindakan yang ditolak, AWS menolak seluruh permintaan dan menghentikan evaluasi. Ini disebut sebagai penolakan secara tegas. Karena permintaan ditolak secara default, AWS mengizinkan permintaan Anda hanya jika setiap bagian dari permintaan Anda diizinkan oleh

kebijakan izin yang berlaku. Logika evaluasi untuk permintaan dalam satu akun mengikuti aturan umum 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 Organizations SCP, batas izin IAM, atau kebijakan sesi membatalkan izin tersebut. Jika ada satu atau lebih jenis kebijakan ini, maka semuanya harus mengizinkan permintaan tersebut. Jika tidak, artinya permintaan ditolak secara implisit.
- Penolakan eksplisit dalam kebijakan mengesampingkan izin apa pun.

Untuk mempelajari lebih lanjut tentang cara semua jenis kebijakan dievaluasi, lihat [Logika evaluasi kebijakan](#). Jika Anda perlu mengajukan permintaan di akun yang berbeda, kebijakan di akun lain harus mengizinkan Anda mengakses sumber daya dan entitas IAM yang Anda gunakan untuk mengajukan permintaan harus memiliki kebijakan berbasis identitas yang mengizinkan permintaan tersebut.

Tindakan atau operasi

Setelah permintaan Anda diautentikasi dan diotorisasi, AWS menyetujui tindakan atau operasi dalam permintaan Anda. Operasi didefinisikan oleh layanan, dan mencakup hal-hal yang dapat Anda lakukan terhadap sumber daya, seperti melihat, membuat, mengedit, dan menghapus sumber daya tersebut. Misalnya, IAM mendukung sekitar 40 tindakan untuk satu sumber daya pengguna, termasuk tindakan berikut:

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

Agar penanggung jawab dapat melakukan operasi, Anda harus menyertakan tindakan yang diperlukan dalam kebijakan yang berlaku bagi prinsipiel atau sumber daya yang terkena dampak. Untuk melihat daftar tindakan, jenis sumber daya, dan kunci syarat yang didukung oleh setiap layanan, lihat [Tindakan, Sumber Daya, dan Kunci Syarat untuk Layanan AWS](#).

Sumber daya

Setelah AWS menyetujui operasi dalam permintaan Anda, operasi dapat dilakukan pada sumber daya terkait di dalam akun Anda. Sumber daya adalah objek yang ada di dalam layanan. Contohnya meliputi instans Amazon EC2, pengguna IAM, dan bucket Amazon S3. Layanan memiliki definisi serangkaian tindakan yang dapat dilakukan pada setiap sumber daya. Jika Anda membuat permintaan untuk melakukan tindakan yang tidak terkait dengan sumber daya, permintaan tersebut ditolak. Misalnya, jika Anda meminta untuk menghapus peran IAM, tetapi memberikan sumber daya grup IAM, permintaan tersebut gagal. Untuk melihat tabel layanan AWS yang mengidentifikasi sumber daya mana yang terdampak oleh suatu tindakan, lihat [Tindakan, Sumber Daya, dan Kunci Syarat untuk Layanan AWS](#).

Gambaran umum manajemen identitas AWS: Pengguna

Anda dapat memberikan akses ke pengguna tertentu dan memberi mereka izin khusus untuk mengakses sumber daya di Akun AWS Anda. Akun AWS Anda dapat menggunakan IAM dan AWS IAM Identity Center untuk membuat pengguna baru atau menggabungkan pengguna yang sudah ada ke dalam. Perbedaan utama antara keduanya adalah bahwa pengguna IAM diberikan kredensial jangka panjang ke AWS sumber daya Anda sementara pengguna di IAM Identity Center memiliki kredensial sementara yang ditetapkan setiap kali pengguna masuk. AWS sebagai [praktik terbaik](#), mewajibkan pengguna manusia untuk menggunakan federasi dengan penyedia identitas untuk mengakses AWS menggunakan kredensial sementara alih-alih sebagai pengguna IAM. Penggunaan utama bagi pengguna IAM adalah untuk memberikan beban kerja yang tidak dapat menggunakan peran IAM kemampuan untuk membuat permintaan terprogram ke AWS layanan menggunakan API atau CLI.

Topik

- [Hanya untuk akses pertama kali: Kredensial pengguna akar Anda](#)
- [Pengguna dan pengguna IAM di IAM Identity Center](#)
- [Menggabungkan pengguna yang ada](#)
- [Metode kontrol akses](#)

Hanya untuk akses pertama kali: Kredensial pengguna akar Anda

Saat Anda membuat Akun AWS, Anda memulai dengan satu identitas masuk yang memiliki akses penuh ke semua Layanan AWS dan sumber daya di akun tersebut. Identitas ini disebut pengguna

root Akun AWS dan diakses dengan cara masuk menggunakan alamat email dan kata sandi yang Anda gunakan untuk membuat akun. Kami sangat menyarankan agar Anda tidak menggunakan pengguna root untuk tugas sehari-hari Anda. Lindungi kredensial pengguna root Anda dan gunakan kredensial tersebut untuk melakukan tugas yang hanya dapat dilakukan pengguna root. Untuk daftar tugas lengkap yang mengharuskan Anda masuk sebagai pengguna root, lihat [Tugas yang memerlukan kredensial pengguna root](#) dalam Panduan Pengguna IAM. Hanya kebijakan kontrol layanan (SCP) dalam organisasi yang dapat membatasi izin yang diberikan kepada pengguna root.

Pengguna dan pengguna IAM di IAM Identity Center

Pengguna IAM bukan akun terpisah; mereka adalah pengguna di dalam akun Anda. Setiap pengguna dapat memiliki kata sandi tersendiri untuk mengakses AWS Management Console. 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 diberikan kredensial jangka panjang ke sumber daya Anda. AWS Sebagai praktik terbaik, jangan membuat pengguna IAM dengan kredensial jangka panjang untuk pengguna manusia Anda. Sebagai gantinya, mintalah pengguna manusia Anda untuk menggunakan kredensial sementara saat mengakses AWS.

Note

Untuk skenario di mana Anda memerlukan pengguna IAM dengan akses terprogram dan kredensial jangka panjang, kami sarankan Anda memperbarui kunci akses bila diperlukan. Untuk informasi selengkapnya, lihat [Memperbarui kunci akses](#).

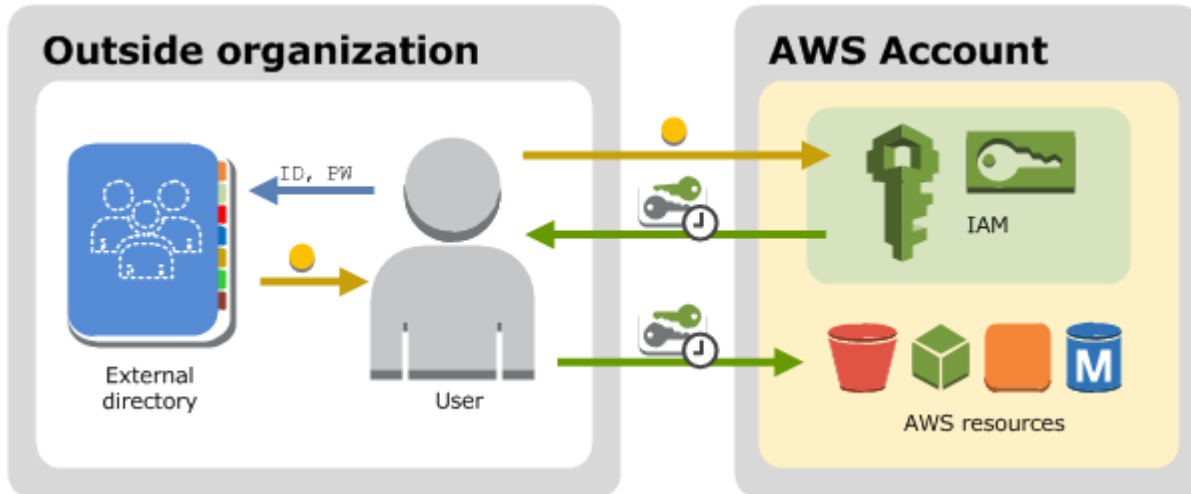
Sebaliknya, pengguna di Pusat Identitas AWS IAM diberikan kredensial jangka pendek untuk sumber daya Anda AWS. Untuk manajemen akses terpusat, kami sarankan Anda menggunakan [AWS IAM Identity Center \(IAM Identity Center\)](#) untuk mengelola akses ke akun Anda dan izin dalam akun tersebut. Pusat Identitas IAM secara otomatis dikonfigurasi dengan direktori Pusat Identitas sebagai sumber identitas default tempat Anda dapat membuat pengguna dan grup, dan menetapkan tingkat akses mereka ke sumber daya Anda AWS. Untuk informasi selengkapnya, lihat [Apa itu AWS IAM Identity Center](#) dalam Panduan Pengguna AWS IAM Identity Center.

Menggabungkan pengguna yang ada

Jika pengguna di organisasi Anda sudah memiliki cara untuk diautentikasi, seperti dengan masuk ke jaringan perusahaan, Anda tidak perlu membuat pengguna IAM atau pengguna terpisah di Pusat

Identitas IAM untuk mereka. Sebagai gantinya, Anda dapat menggabungkan identitas pengguna tersebut ke dalam AWS menggunakan IAM atau AWS IAM Identity Center

Diagram berikut menunjukkan bagaimana pengguna bisa mendapatkan kredensial AWS keamanan sementara untuk mengakses sumber daya di Anda. Akun AWS



Federasi sangat 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 memberikan akses masuk tunggal (SSO) ke AWS Management Console untuk pengguna Anda. 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 memberikan akses masuk tunggal (SSO) ke AWS Management Console untuk pengguna Anda. Untuk informasi selengkapnya, lihat [Mengaktifkan akses broker identitas khusus ke konsol AWS](#).

Jika direktori perusahaan Anda adalah Microsoft Active Directory, Anda dapat menggunakan 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 menggunakan AWS IAM Identity Center untuk membangun

kepercayaan antara IDP Anda dan Anda. Akun AWS Untuk informasi selengkapnya, lihat [Connect 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 kontrol akses

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

Jenis akses pengguna	Mengapa saya harus menggunakannya?	Di mana saya bisa mendapatkan informasi lebih lanjut?
Akses masuk tunggal untuk pengguna manusia, seperti pengguna tenaga kerja Anda, ke AWS sumber daya menggunakan IAM Identity Center	<p>IAM Identity Center menyediakan tempat sentral yang menyatukan administrasi pengguna dan akses mereka ke Akun AWS dan aplikasi cloud.</p> <p>Anda dapat mengatur penyimpanan identitas dalam IAM Identity Center atau Anda dapat mengonfigurasi federasi dengan penyedia identitas yang ada (iDP). Memberikan kredensi terbatas kepada pengguna manusia Anda</p>	<p>Untuk informasi selengkapnya tentang menyiapkan Pusat Identitas IAM, lihat Memulai di AWS IAM Identity CenterPanduan Pengguna</p> <p>Untuk informasi selengkapnya tentang penggunaan MFA di Pusat Identitas IAM, lihat Autentikasi multi-faktor di Panduan Pengguna AWS IAM Identity Center</p>

Jenis akses pengguna	Mengapa saya harus menggunakannya?	Di mana saya bisa mendapatkan informasi lebih lanjut?
	<p>untuk AWS sumber daya sesuai kebutuhan direkomendasikan sebagai praktik terbaik keamanan.</p> <p>Pengguna 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.</p>	
Akses gabungan untuk pengguna manusia, seperti pengguna tenaga kerja Anda, ke AWS layanan yang menggunakan penyedia identitas IAM	Dukungan IAM IdPs yang kompatibel dengan OpenID Connect (OIDC) atau SAMP 2.0 (Security Assertion Markup Language 2.0). Setelah membuat penyedia identitas IAM, Anda harus membuat satu atau beberapa peran IAM yang dapat ditetapkan secara dinamis ke pengguna federasi.	Untuk informasi lebih lanjut tentang penyedia identitas dan federasi IAM, lihat Penyedia dan federasi identitas .

Jenis akses pengguna	Mengapa saya harus menggunakannya?	Di mana saya bisa mendapatkan informasi lebih lanjut?
Akses lintas akun antara Akun AWS	<p>Anda ingin berbagi akses ke AWS sumber daya tertentu dengan pengguna lainAkun AWS.</p> <p>Peran adalah cara utama untuk memberikan akses akun silang. Namun, beberapa layanan AWS memungkinkan Anda melampirkan kebijakan secara langsung ke sumber daya (alih-alih menggunakan peran sebagai proksi). Ini disebut kebijakan berbasis sumber daya.</p>	<p>Untuk informasi selengkapnya tentang peran IAM, lihat Peran IAM.</p> <p>Untuk informasi selengkapnya tentang peran terkait layanan, lihat Menggunakan peran terkait layanan.</p> <p>Untuk informasi tentang layanan mana yang mendukung penggunaan peran terkait layanan, lihat AWS layanan yang bekerja dengan IAM Cari layanan yang memiliki Ya di kolom Peran Bertaut Layanan. Untuk melihat dokumentasi peran terkait layanan untuk layanan tersebut, pilih tautan yang terkait dengan Ya di kolom tersebut.</p>

Jenis akses pengguna	Mengapa saya harus menggunakannya?	Di mana saya bisa mendapatkan informasi lebih lanjut?
<p>Kredensi jangka panjang untuk pengguna IAM yang ditunjuk di Anda Akun AWS</p>	<p>Anda mungkin memiliki kasus penggunaan khusus yang memerlukan kredensial jangka panjang dengan pengguna IAM di. AWS Anda dapat menggunakan IAM untuk membuat pengguna IAM ini di Anda Akun AWS, dan menggunakan IAM untuk mengelola izin mereka. Beberapa kasus penggunaan meliputi:</p> <ul style="list-style-type: none"> • Beban kerja yang tidak dapat menggunakan peran IAM • AWSKlien pihak ketiga yang memerlukan akses terprogram melalui kunci akses • Kredensi khusus layanan untuk atau Amazon Keyspaces AWS CodeCommit • AWS IAM Identity Center tidak tersedia untuk akun Anda dan Anda tidak memiliki penyedia identitas lain <p>Sebagai praktik terbaik dalam skenario di mana Anda memerlukan pengguna IAM dengan akses terprogra</p>	<p>Untuk informasi selengkapnya tentang pengaturan pengguna IAM, lihat Membuat pengguna IAM di Akun AWS.</p> <p>Untuk informasi selengkapnya tentang kunci akses pengguna IAM, lihat Mengelola kunci akses untuk pengguna IAM.</p> <p>Untuk informasi selengkapnya tentang kredensial khusus layanan untuk atau AWS CodeCommit Amazon Keyspaces, lihat dan Menggunakan IAM dengan CodeCommit: Kredensial Git, kunci SSH, dan kunci akses AWS Menggunakan IAM dengan Amazon Keyspaces (untuk Apache Cassandra)</p>

Jenis akses pengguna	Mengapa saya harus menggunakannya?	Di mana saya bisa mendapatkan informasi lebih lanjut?
	m dan kredensi jangka panjang , kami menyarankan Anda memperbarui kunci akses saat diperlukan. Untuk informasi selengkapnya, lihat Memperbarui kunci akses .	

Gambaran umum manajemen akses: Izin dan kebijakan

Bagian manajemen akses AWS Identity and Access Management (IAM) membantu Anda menentukan entitas utama yang diizinkan untuk dilakukan di akun. Entitas penanggung jawab adalah orang atau aplikasi yang diautentikasi menggunakan entitas IAM (pengguna atau peran). Manajemen akses sering disebut sebagai otorisasi. Anda mengelola akses AWS dengan membuat kebijakan dan melampirkannya ke identitas IAM (pengguna, grup pengguna, atau peran) atau sumber daya. AWS Kebijakan adalah objek AWS yang, ketika dikaitkan dengan identitas atau sumber daya, mendefinisikan izin mereka. AWS mengevaluasi kebijakan ini ketika prinsipal menggunakan entitas IAM (pengguna atau peran) untuk membuat permintaan. Izin dalam kebijakan dapat menentukan permintaan yang diizinkan atau ditolak. Sebagian besar kebijakan disimpan dalam AWS sebagai dokumen JSON. Untuk informasi selengkapnya tentang tipe dan penggunaan kebijakan, lihat [Kebijakan dan Izin di IAM](#).

Kebijakan dan akun

Jika Anda mengelola satu akun dalam AWS, Anda menentukan izin di dalam akun tersebut menggunakan kebijakan. Jika Anda mengelola izin di beberapa akun, akan lebih sulit untuk mengelola izin bagi pengguna Anda. Anda dapat menggunakan peran IAM, kebijakan berbasis sumber daya, atau access control list (ACL) untuk izin lintas akun. Namun, jika Anda memiliki beberapa akun, sebaiknya gunakan layanan AWS Organizations untuk membantu Anda mengelola izin tersebut. Untuk informasi lebih lanjut, lihat [Apa itu AWS Organizations?](#) dalam Panduan Pengguna Organisasi.

Kebijakan dan pengguna

Pengguna IAM adalah identitas dalam layanan. 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 dengan membuat kebijakan berbasis identitas, yang merupakan kebijakan yang dilampirkan pada pengguna atau grup tempat pengguna berada. Contoh berikut menunjukkan kebijakan JSON yang memungkinkan pengguna untuk melakukan semua tindakan Amazon DynamoDB (`dynamodb:*`) pada Books tabel di 123456789012 akun dalam Wilayah. `us-east-2`

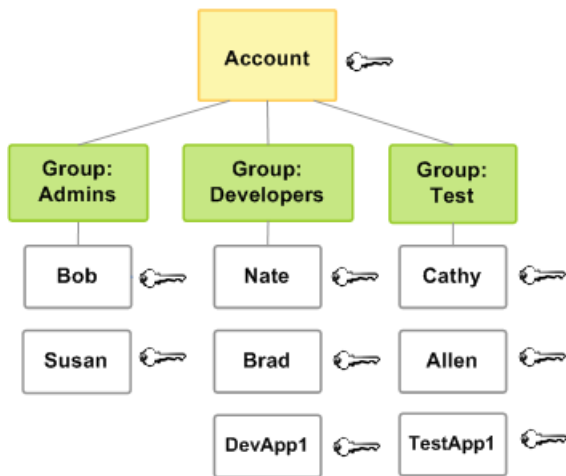
```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "dynamodb:*",
    "Resource": "arn:aws:dynamodb:us-east-2:123456789012:table/Books"
  }
}
```

Setelah Anda melampirkan kebijakan ini ke pengguna IAM Anda, pengguna hanya memiliki izin DynamoDB tersebut. Sebagian besar pengguna memiliki beberapa kebijakan yang bersama-sama mewakili izin bagi pengguna tersebut.

Tindakan atau sumber daya yang tidak secara eksplisit diizinkan akan ditolak secara default. Misalnya, jika kebijakan sebelumnya adalah satu-satunya kebijakan yang dilampirkan ke pengguna, maka pengguna tersebut diizinkan untuk hanya melakukan tindakan DynamoDB di atas meja. Books Tindakan di semua tabel lainnya dilarang. Demikian pula, pengguna tidak diizinkan untuk melakukan tindakan apa pun di Amazon EC2, Amazon S3, atau di layanan lainnya AWS. Alasannya adalah bahwa izin untuk bekerja dengan layanan tersebut tidak termasuk dalam kebijakan.

Kebijakan dan grup

Anda dapat menata pengguna IAM menjadi Grup IAM dan melampirkan kebijakan ke grup. Dalam hal ini, pengguna individu masih memiliki kredensial mereka sendiri, tetapi semua pengguna dalam grup memiliki izin yang dilampirkan ke grup. Gunakan grup untuk pengelolaan izin yang lebih mudah, dan untuk mengikuti kam [Praktik terbaik keamanan di IAM](#).



Pengguna atau grup dapat memiliki beberapa kebijakan yang dilampirkan pada mereka yang memberikan izin yang berbeda. Dalam hal ini, izin untuk pengguna dihitung berdasarkan kombinasi kebijakan. Namun, prinsip dasar masih berlaku: Jika pengguna belum diberi izin secara eksplisit untuk tindakan dan sumber daya, pengguna tidak memiliki izin tersebut.

Pengguna gabungan dan peran

Pengguna federasi tidak memiliki identitas permanen seperti yang dilakukan Akun AWS pengguna IAM. Untuk menetapkan izin bagi pengguna gabungan, Anda dapat membuat entitas yang disebut sebagai peran dan menentukan izin untuk peran tersebut. Ketika pengguna gabungan masuk ke AWS, pengguna tersebut dikaitkan dengan peran dan diberi izin yang didefinisikan dalam peran. Untuk informasi selengkapnya, lihat [Membuat peran untuk Penyedia Identitas pihak ketiga \(federasi\)](#).

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. Akun AWS Anda dapat menggunakan dua tipe kebijakan terkelola:

- **AWSkebijakan terkelola** — Kebijakan terkelola yang dibuat dan dikelola olehAWS. Jika Anda baru menggunakan kebijakan, sebaiknya mulai dengan menggunakan kebijakan yang dikelola AWS.
- **Kebijakan yang dikelola pelanggan** — Kebijakan terkelola yang Anda buat dan kelola dalam kebijakan AndaAkun AWS. Kebijakan yang dikelola pelanggan memberikan kontrol yang lebih tepat atas kebijakan Anda dibandingkan kebijakan yang dikelola AWS. Anda dapat membuat, mengedit, dan memvalidasi kebijakan IAM di editor visual atau dengan membuat dokumen kebijakan JSON secara langsung. Untuk informasi lebih lanjut, lihat [Membuat kebijakan IAM](#) dan [Menyunting 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 penanggung jawab kebijakan berbasis sumber daya.

Layanan IAM hanya mendukung satu jenis kebijakan berbasis sumber daya yang disebut kebijakan kepercayaan peran, yang terlampir pada peran IAM. Karena peran IAM merupakan sebuah identitas dan sumber daya yang mendukung kebijakan berbasis sumber daya, Anda harus melampirkan kebijakan kepercayaan dan kebijakan berbasis identitas pada peran IAM. Kebijakan kepercayaan menentukan entitas penanggung jawab 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 layanan mana yang mendukung kebijakan berbasis sumber daya, lihat [AWS layanan yang bekerja dengan IAM](#). Untuk mempelajari selengkapnya tentang kebijakan berbasis sumber daya, lihat [Kebijakan berbasis identitas dan kebijakan berbasis sumber daya](#).

Untuk apa ABAC? AWS

Kontrol akses berbasis atribut (ABAC) adalah strategi otorisasi yang menentukan izin berdasarkan atribut. Dalam AWS, atribut ini disebut tag. Anda dapat melampirkan tag ke sumber daya IAM, termasuk entitas IAM (pengguna atau peran) dan sumber daya. AWS Anda dapat membuat kebijakan

ABAC tunggal atau set kecil kebijakan untuk penanggung jawab IAM Anda. Kebijakan ABAC ini dapat dirancang untuk memungkinkan operasi ketika tag penanggung jawab cocok dengan tag sumber daya. ABAC sangat membantu di lingkungan yang berkembang dengan cepat dan membantu dalam situasi ketika manajemen kebijakan menjadi rumit.

Misalnya, Anda dapat membuat tiga peran dengan kunci tanda `access-project`. Mengatur nilai tanda dari peran pertama ke `Heart`, yang kedua ke `Star`, dan yang ketiga ke `Lightning`. Anda kemudian dapat menggunakan kebijakan tunggal yang memungkinkan akses ketika peran dan sumber daya ditandai dengan nilai yang sama untuk `access-project`. Untuk tutorial terperinci yang menunjukkan cara menggunakan ABAC di AWS, lihat [Tutorial IAM: Menentukan izin untuk mengakses sumber daya AWS berdasarkan tanda](#) Untuk mempelajari tentang layanan yang mendukung ABAC, lihat [AWS layanan yang bekerja dengan IAM](#).

Membandingkan ABAC dengan model RBAC tradisional

Model otorisasi tradisional yang digunakan di IAM disebut kontrol akses berbasis peran (RBAC). RBAC mendefinisikan izin berdasarkan fungsi tugas seseorang, yang dikenal di luar AWS sebagai peran. Dalam AWS peran biasanya mengacu pada peran IAM, yang merupakan identitas dalam IAM yang dapat Anda asumsikan. 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 pengguna, atau peran IAM). Sebagai [praktik terbaik](#), Anda memberikan izin minimum yang diperlukan untuk fungsi pekerjaan. Ini dikenal sebagai [memberikan hak istimewa terkecil](#). Lakukan ini dengan mencantumkan sumber daya tertentu yang dapat diakses oleh fungsi pekerjaan. Kekurangan menggunakan model RBAC tradisional ini adalah saat karyawan menambahkan sumber daya baru, Anda harus memperbarui kebijakan untuk memungkinkan 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 diizinkan siapa pun untuk menggunakan peran yang dapat diakses. Jika seorang karyawan berganti pekerjaan dalam perusahaan Anda, Anda menetapkan mereka ke peran IAM yang berbeda. Orang atau program dapat ditetapkan ke lebih dari satu peran. Namun, `Star` proyek ini mungkin memerlukan sumber daya tambahan, seperti wadah Amazon EC2 baru. Dalam hal ini, Anda harus memperbarui kebijakan yang dilampirkan ke `Star` peran untuk menentukan sumber daya penampung baru. Jika tidak, anggota `Star` proyek tidak diizinkan mengakses kontainer baru.

ABAC memberikan keuntungan berikut dibandingkan model RBAC tradisional:

- Skala izin ABAC dengan inovasi. Administrator tidak perlu lagi memperbarui kebijakan yang ada untuk memungkinkan akses ke sumber daya baru. Misalnya, anggap Anda merancang strategi ABAC Anda dan tanda `access-project`. Developer menggunakan peran dengan tanda `access-project = Heart`. 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.
- Menggunakan ABAC, tim dapat berubah dan tumbuh dengan cepat. Ini karena izin untuk sumber daya baru secara otomatis diberikan berdasarkan atribut. Misalnya, jika perusahaan Anda sudah mendukung proyek `Heart` dan `Star` menggunakan ABAC, mudah untuk menambahkan proyek `Lightning` baru. Administrator IAM membuat peran baru dengan tag `access-project = Lightning`. Tidak perlu mengubah kebijakan untuk mendukung proyek baru. Siapa pun yang memiliki izin untuk mengasumsikan peran dapat membuat dan melihat instans yang ditandai dengan `access-project = Lightning`. Selain itu, anggota tim dapat berpindah dari proyek `Heart` ke proyek `Lightning`. Administrator IAM menugaskan pengguna 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 harus menyusun kebijakan yang memungkinkan akses hanya ke sumber daya tertentu. Namun, ketika Anda menggunakan ABAC, Anda dapat mengizinkan tindakan pada semua sumber daya, tetapi hanya jika tanda sumber daya cocok dengan tanda prinsipal.
- Gunakan atribut karyawan dari direktori perusahaan Anda dengan ABAC. Anda dapat mengonfigurasi penyedia SAML atau OIDC Anda untuk meneruskan tag sesi ke AWS. Ketika karyawan Anda bergabung AWS, atribut mereka diterapkan pada AWS prinsipal 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: Menentukan izin untuk mengakses sumber daya AWS berdasarkan tanda](#)

Fitur keamanan di luar IAM

Anda menggunakan IAM untuk mengontrol akses ke tugas yang dilakukan AWS Management Console [menggunakan—Alat Baris AWS Perintah](#), atau operasi API layanan menggunakan [AWS SDK](#). 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:

- [Memulai dengan Instans Linux Amazon EC2](#) di Panduan Pengguna Amazon EC2 untuk Instans Linux
- [Memulai dengan Instans Windows Amazon EC2](#) di Panduan Pengguna Amazon EC2 untuk Instans Windows

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 suatu instans atau basis data.

Untuk informasi selengkapnya, lihat dokumentasi berikut:

- [Grup Keamanan Amazon EC2 untuk Instans Linux](#) di Panduan Pengguna Amazon EC2 untuk Instans Linux
- [Grup Keamanan Amazon EC2 untuk Instans Windows](#) di Panduan Pengguna Amazon EC2 untuk Instans Windows
- [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.

Amazon WorkDocs

Di Amazon WorkDocs, pengguna mendapatkan akses ke dokumen bersama dengan masuk menggunakan nama pengguna dan kata sandi.

Untuk informasi selengkapnya, lihat [Memulai dengan Amazon WorkDocs](#) di Panduan WorkDocs Administrasi Amazon.

Metode kontrol akses ini bukan bagian dari IAM. IAM memungkinkan Anda mengontrol bagaimana AWS produk ini dikelola—membuat atau menghentikan instans Amazon EC2, menyiapkan WorkSpaces desktop baru, dan sebagainya. Yaitu, IAM membantu Anda mengontrol tugas yang dilakukan dengan membuat permintaan ke Amazon Web Services, dan membantu Anda mengontrol akses ke AWS Management Console. Namun, IAM tidak membantu Anda mengelola keamanan untuk tugas seperti masuk ke sistem operasi (Amazon EC2), basis data (Amazon RDS), desktop (Amazon WorkSpaces), atau situs kolaborasi (Amazon WorkDocs).

Ketika Anda bekerja dengan produk AWS tertentu, pastikan Anda membaca dokumentasi untuk mempelajari opsi keamanan untuk semua sumber daya yang menjadi milik produk tersebut.

Tautan cepat ke tugas umum

Gunakan tautan berikut untuk mendapatkan bantuan tentang tugas umum yang terkait dengan IAM.

Masuk untuk berbagai jenis pengguna

Masuklah ke [Konsol IAM](#) dengan memilih Pengguna IAM dan masukkan akun alias atau ID Akun AWS Anda. Di halaman berikutnya, masukkan nama pengguna IAM dan kata sandi Anda.

Untuk masuk dengan pengguna Pusat Identitas IAM, gunakan URL masuk yang dikirim ke alamat email Anda saat Anda membuat pengguna Pusat Identitas IAM.

Untuk bantuan masuk menggunakan pengguna Pusat Identitas IAM, lihat [Masuk ke portal akses AWS](#) dalam Panduan Pengguna AWS Sign-In.

Masuk ke [AWS Management Console](#) sebagai pemilik akun dengan memilih Pengguna root dan memasukkan alamat email Akun AWS Anda. Pada halaman berikutnya, masukkan kata sandi Anda.

Lihat [Apa itu AWS Masuk](#) di Panduan AWS Sign-In Pengguna untuk bantuan menentukan jenis pengguna dan halaman login Anda.

Kelola kata sandi untuk pengguna

Anda memerlukan kata sandi untuk mengakses AWS Management Console, termasuk akses ke informasi penagihan.

Untuk Anda Pengguna root akun AWS, lihat [Mengubah kata sandi untuk Pengguna root akun AWS di](#) Panduan AWS Account Management Referensi

Untuk pengguna IAM, lihat [Mengelola kata sandi untuk pengguna IAM](#).

Kelola izin untuk pengguna

Anda menggunakan kebijakan untuk memberikan izin kepada pengguna IAM di Anda. Akun AWS Pengguna IAM tidak memiliki izin saat dibuat, jadi Anda harus menambahkan izin untuk memungkinkan mereka menggunakan sumber daya. AWS

Untuk memberikan akses, tambahkan izin ke pengguna, grup, atau peran Anda:

- Pengguna dan grup di AWS IAM Identity Center:

Buat set izin. Ikuti instruksi di [Buat set izin](#) di Panduan Pengguna AWS IAM Identity Center.

- Pengguna yang dikelola di IAM melalui penyedia identitas:

Buat peran untuk federasi identitas. Ikuti instruksi dalam [Membuat peran untuk penyedia identitas pihak ketiga \(federasi\)](#) di Panduan Pengguna IAM.

- Pengguna IAM:
 - Buat peran yang dapat diambil pengguna Anda. Ikuti instruksi dalam [Membuat peran untuk pengguna IAM](#) dalam Panduan Pengguna IAM.
 - (Tidak disarankan) Lampirkan kebijakan langsung ke pengguna atau tambahkan pengguna ke grup pengguna. Ikuti instruksi dalam [Menambahkan izin ke pengguna \(konsol\)](#) dalam Panduan Pengguna IAM.

Untuk informasi selengkapnya, lihat [Mengelola kebijakan IAM](#).

Buat daftar pengguna di Anda Akun AWS dan dapatkan informasi tentang kredensialnya

Lihat [Mendapatkan laporan kredensi untuk Anda Akun AWS](#).

Tambahkan multi-factor authentication (MFA)

Untuk menambahkan perangkat MFA virtual, lihat salah satu hal berikut:

- [Aktifkan perangkat MFA virtual untuk Pengguna root akun AWS \(konsol\) Anda](#)
- [Aktifkan perangkat MFA virtual untuk pengguna IAM \(konsol\)](#)

Untuk menambahkan kunci keamanan FIDO, lihat salah satu dari berikut ini:


- [Aktifkan kunci keamanan FIDO untuk pengguna Akun AWS root \(konsol\)](#)
- [Aktifkan kunci keamanan FIDO untuk pengguna IAM lain \(konsol\)](#)

Untuk menambahkan perangkat keras MFA, lihat salah satu hal berikut:

- [Aktifkan token TOTP perangkat keras untuk pengguna Akun AWS root \(konsol\)](#).
- [Aktifkan token TOTP perangkat keras untuk pengguna IAM lain \(konsol\)](#)

Dapatkan access key

Anda dapat menggunakan kunci akses untuk membuat AWS permintaan menggunakan [AWSSDK](#), [Alat Baris AWS Perintah](#), atau operasi API.

 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](#).

Untuk panduan untuk membantu Anda melindungi kunci akses, lihat [Mengamankan kunci akses](#).

Untuk mempelajari tentang mengelola kunci akses bagi pengguna IAM, lihat [Mengelola kunci akses untuk pengguna IAM](#).

Untuk informasi selengkapnya tentang kredensial keamanan yang tersedia untuk Anda Akun AWS, lihat kredensi [AWSkeamanan](#).

Tag sumber daya IAM

Anda dapat menandai sumber daya IAM berikut:

- Pengguna IAM
- Peran IAM
- Kebijakan yang dikelola pelanggan
- Penyedia Identitas
- Sertifikat server
- Perangkat MFA virtual

Untuk mempelajari tentang tag di IAM, lihat [Menandai sumber daya IAM](#).

Untuk mempelajari tentang penggunaan tanda untuk mengontrol akses ke sumber daya AWS, lihat [Mengontrol akses ke AWS sumber daya menggunakan tag](#).

Lihat tindakan, sumber daya, dan kunci kondisi untuk semua layanan

Seperangkat dokumentasi referensi ini dapat membantu Anda menulis kebijakan IAM terperinci. Setiap AWS layanan menentukan tindakan, sumber daya, dan kunci konteks kondisi yang Anda gunakan dalam kebijakan IAM. Untuk mempelajari selengkapnya, lihat [Tindakan, Sumber Daya, dan Kunci Kondisi untuk Layanan AWS](#).

Memulai dengan keseluruhan AWS

Rangkaian dokumentasi ini utamanya berkaitan dengan layanan IAM. Untuk mempelajari tentang memulai dengan AWS dan menggunakan beberapa layanan untuk memecahkan masalah seperti membangun dan meluncurkan proyek pertama Anda, lihat [Memulai Pusat Sumber Daya](#).

Pencarian konsol IAM

Gunakan halaman pencarian konsol IAM sebagai opsi yang lebih cepat untuk menemukan sumber daya IAM. Anda dapat menggunakan pencarian konsol untuk menemukan kunci akses yang terkait dengan akun Anda, entitas IAM (seperti pengguna, grup, peran, penyedia identitas), kebijakan berdasarkan nama, dan lainnya.

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 menyetikkan 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.

Menggunakan pencarian konsol IAM



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 Pengguna AWS Masuk.
2. Pada halaman Beranda Konsol, pilih layanan 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.

Ikun di hasil pencarian konsol IAM

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

Ikun	Deskripsi
	Pengguna IAM:
	Grup IAM

Ikon	Deskripsi
	IAM role
	Kebijakan IAM
	Tugas seperti “buat pengguna” atau “lampirkan kebijakan”
	Hasil dari kata kunci delete

Contoh frasa 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*

Menciptakan AWS Identity and Access Management sumber daya dengan AWS 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. AWS CloudFormation

Bila Anda menggunakan AWS 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 AWS CloudFormation

Untuk menyediakan dan mengonfigurasi sumber daya untuk IAM dan layanan terkait, Anda harus memahami [AWS CloudFormation templat](#). Templat adalah file teks dengan format JSON atau YAML. Template ini menjelaskan sumber daya yang ingin Anda sediakan di AWS CloudFormation tumpukan Anda. Jika Anda tidak terbiasa dengan JSON atau YAMM, Anda dapat menggunakan AWS CloudFormation Designer untuk membantu Anda memulai dengan template. AWS CloudFormation Lihat informasi selengkapnya di [Apa yang dimaksud dengan Desainer AWS CloudFormation ?](#) dalam Panduan Pengguna AWS CloudFormation .

IAM mendukung pembuatan kunci akses, grup, kebijakan grup, profil instans, kebijakan terkelola, penyedia OIDC, kebijakan inline, peran, kebijakan peran, penyedia SAML, sertifikat server, peran terkait layanan, pengguna (dan menambahkan pengguna ke grup), kebijakan pengguna, dan perangkat MFA virtual di AWS CloudFormation Untuk informasi selengkapnya, termasuk contoh template JSON dan YAML 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 AWS CloudFormation

Untuk mempelajari selengkapnya AWS CloudFormation, lihat sumber daya berikut:

- [AWS CloudFormation](#)
- [AWS CloudFormation Panduan Pengguna](#)
- [AWS CloudFormation Referensi API](#)
- [AWS CloudFormation Panduan Pengguna Antarmuka Baris Perintah](#)

Menggunakan AWS CloudShell untuk bekerja dengan AWS Identity and Access Management

AWS CloudShell adalah shell pra-otentikasi berbasis browser yang dapat Anda luncurkan langsung dari file. AWS Management Console Anda dapat menjalankan AWS CLI perintah terhadap AWS layanan (termasuk AWS Identity and Access Management) 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 AWS Management Console](#), 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 pada lingkungan komputasi shell).

Memperoleh izin IAM untuk AWS CloudShell

Dengan 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 ke pengguna adalah melalui kebijakan AWS terkelola. [Kebijakan AWS terkelola](#) adalah kebijakan mandiri yang dibuat dan dikelola oleh AWS. Kebijakan AWS terkelola berikut ini CloudShell dapat dilampirkan ke identitas IAM:

- `AWSCloudShellFullAccess`: Memberikan izin untuk menggunakan AWS CloudShell dengan akses penuh ke semua fitur.

Jika ingin membatasi cakupan tindakan yang dapat dilakukan oleh pengguna IAM AWS CloudShell, Anda dapat membuat kebijakan kustom yang menggunakan kebijakan `AWSCloudShellFullAccess` terkelola sebagai templat. Untuk informasi selengkapnya tentang membatasi tindakan yang tersedia bagi pengguna CloudShell, lihat [Mengelola AWS CloudShell akses dan penggunaan dengan kebijakan IAM](#) di Panduan AWS CloudShell Pengguna.

Berinteraksi dengan IAM menggunakan AWS CloudShell

Setelah Anda meluncurkan AWS CloudShell dari AWS Management Console, 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 AWS Management Console, Anda dapat meluncurkan CloudShell dengan memilih opsi berikut yang tersedia di bilah navigasi:

- Pilih CloudShell ikon.
 - Mulai mengetik “cloudshell” di kotak Pencarian dan kemudian pilih opsi. CloudShell
2. Untuk membuat grup IAM, masukkan perintah berikut di baris CloudShell perintah. Dalam contoh ini kami 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 kami menamai grup `east_coast` dan pengguna `johndoe`:

```
aws iam add-user-to-group --group-name east_coast --user-name johndoe
```

4. Untuk memverifikasi bahwa pengguna berada 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": "johndoe",
  "UserId": "AIDAYBDBW4JBXGEXAMPLE",
  "Arn": "arn:aws:iam::552108220995:user/johndoe",
  "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 IAM dengan SDK AWS

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 for C++	AWS SDK for C++ contoh kode
AWS SDK for Go	AWS SDK for Go contoh kode
AWS SDK for Java	AWS SDK for Java contoh kode
AWS SDK for JavaScript	AWS SDK for JavaScript contoh kode
AWS SDK for Kotlin	AWS SDK for Kotlin contoh kode
AWS SDK for .NET	AWS SDK for .NET contoh kode
AWS SDK for PHP	AWS SDK for PHP contoh kode

Dokumentasi SDK	Contoh kode
AWS SDK for Python (Boto3)	AWS SDK for Python (Boto3) contoh kode
AWS SDK for Ruby	AWS SDK for Ruby contoh kode
AWS SDK for Rust	AWS SDK for Rust contoh kode
AWS SDK untuk SAP ABAP	AWS SDK untuk SAP ABAP contoh kode
AWS SDK for Swift	AWS SDK for Swift contoh kode

Untuk contoh khusus untuk IAM, lihat [Contoh kode untuk IAM menggunakan AWS SDK](#).

 **Ketersediaan contoh**

Tidak menemukan yang Anda cari? Minta contoh kode menggunakan tautan Berikan umpan balik di bagian bawah halaman ini.

Menyiapkan IAM

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.

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 kerahasiaan login Anda. Anda tidak secara khusus mendaftar untuk menggunakan IAM. Tidak ada biaya untuk menggunakan IAM.

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 yang Anda kelola secara eksternal AWS atau Anda dapat membuat pengguna yang sedang AWS menggunakannya 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 AWS IAM Identity Center User Guide.

Note

IAM terintegrasi dengan beberapa AWS produk. Untuk daftar layanan yang mendukung IAM, lihat [AWS layanan yang bekerja dengan IAM](#).

Topik

- [Mendaftar Akun AWS](#)
- [Membuat pengguna administratif](#)
- [Bersiaplah untuk izin dengan hak istimewa paling sedikit](#)
- [Metode manajemen IAM](#)
- [Akun AWS ID Anda dan aliasnya](#)

Mendaftar Akun AWS

Jika Anda tidak memiliki Akun AWS, selesaikan langkah-langkah berikut untuk membuatnya.

Untuk mendaftar Akun AWS

1. Buka <https://portal.aws.amazon.com/billing/signup>.
2. Ikuti petunjuk secara online.

Anda akan diminta untuk menerima panggilan telepon dan memasukkan kode verifikasi pada keypad telepon sebagai bagian dari prosedur pendaftaran.

Saat Anda mendaftar Akun AWS, Pengguna root akun AWS akan dibuat. Pengguna root memiliki akses ke semua Layanan AWS dan sumber daya dalam akun. Sebagai praktik terbaik keamanan, [tetapkan akses administratif ke pengguna administratif](#), dan hanya gunakan pengguna root untuk melakukan [tugas yang memerlukan akses pengguna root](#).

AWS akan mengirimkan email konfirmasi kepada Anda setelah proses pendaftaran selesai. Anda dapat melihat aktivitas akun saat ini dan mengelola akun dengan mengunjungi <https://aws.amazon.com/> dan memilih Akun Saya.

Membuat pengguna administratif

Setelah mendaftar Akun AWS, amankan Pengguna root akun AWS, aktifkan AWS IAM Identity Center, dan buat sebuah pengguna administratif sehingga Anda tidak menggunakan pengguna root untuk tugas sehari-hari.

Mengamankan Pengguna root akun AWS Anda

1. Masuk ke [AWS Management Console](#) sebagai pemilik akun dengan memilih Pengguna root dan memasukkan alamat email Akun AWS Anda. Di halaman berikutnya, masukkan kata sandi Anda.

Untuk bantuan masuk menggunakan pengguna root, lihat [Masuk sebagai pengguna root](#) dalam Panduan Pengguna AWS Sign-In.

2. Aktifkan autentikasi multi-faktor (MFA) untuk pengguna root Anda.

Untuk petunjuknya, silakan lihat [Mengaktifkan perangkat MFA virtual untuk pengguna root Akun AWS Anda \(konsol\)](#) dalam Panduan Pengguna IAM.

Membuat pengguna administratif

1. Aktifkan Pusat Identitas IAM.

Untuk mendapatkan petunjuk, silakan lihat [Mengaktifkan AWS IAM Identity Center](#) di Panduan Pengguna AWS IAM Identity Center.

2. Di Pusat Identitas IAM, berikan akses administratif ke sebuah pengguna administratif.

Untuk mendapatkan tutorial tentang menggunakan Direktori Pusat Identitas IAM sebagai sumber identitas Anda, silakan lihat [Mengonfigurasi akses pengguna dengan Direktori Pusat Identitas IAM default](#) di Panduan Pengguna AWS IAM Identity Center.

Masuk sebagai pengguna administratif

- Untuk masuk dengan pengguna Pusat Identitas IAM, gunakan URL masuk yang dikirim ke alamat email Anda saat Anda membuat pengguna Pusat Identitas IAM.

Untuk bantuan masuk menggunakan pengguna Pusat Identitas IAM, lihat [Masuk ke portal akses AWS](#) dalam Panduan Pengguna AWS Sign-In.

Bersiaplah untuk izin dengan hak istimewa paling sedikit

Menggunakan izin hak istimewa paling rendah adalah rekomendasi praktik terbaik IAM. Konsep izin hak istimewa paling rendah adalah memberi pengguna izin yang diperlukan untuk melakukan tugas dan tidak ada izin tambahan. Saat Anda menyiapkan, pertimbangkan bagaimana Anda akan mendukung izin hak istimewa paling sedikit. Baik pengguna root dan pengguna administrator 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 IAM Identity Center 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 terhubung ke sumber daya tersebut dengan akun istimewa Anda yang lebih rendah.

Jika Anda menggunakan IAM Identity Center, pertimbangkan untuk menggunakan set izin IAM Identity Center untuk memulai. Untuk mempelajari selengkapnya, lihat [Membuat set izin](#) di Panduan Pengguna Pusat Identitas IAM.

Jika Anda tidak menggunakan IAM Identity Center, gunakan peran IAM untuk menentukan izin untuk entitas IAM yang berbeda. Untuk mempelajari selengkapnya, lihat [Membuat Peran IAM](#).

Kedua peran IAM dan kumpulan izin Pusat Identitas IAM dapat menggunakan kebijakan AWS terkelola berdasarkan fungsi pekerjaan. Untuk detail tentang izin yang diberikan oleh kebijakan ini, lihat [AWS kebijakan terkelola untuk fungsi pekerjaan](#).

Important

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 Setelah menyiapkan, kami menyarankan Anda menggunakan IAM Access Analyzer untuk membuat kebijakan hak istimewa paling sedikit berdasarkan aktivitas akses Anda yang masuk. AWS CloudTrail Untuk informasi selengkapnya tentang pembuatan kebijakan, lihat pembuatan [kebijakan IAM Access Analyzer](#).

Metode manajemen IAM

Anda dapat mengelola IAM menggunakan AWS konsol, antarmuka AWS baris perintah, atau melalui antarmuka aplikasi (API) di SDK terkait. Saat Anda bersiap, pertimbangkan metode mana yang ingin Anda dukung dan bagaimana Anda berencana untuk mendukung pengguna yang berbeda.

Topik

- [Konsol AWS](#)
- [AWS Antarmuka Baris Perintah \(CLI\) dan Kit Pengembangan Perangkat Lunak \(SDK\)](#)

Konsol 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, pengguna federasi, dan pengguna di IAM Identity Center memiliki kredensial sementara, sedangkan pengguna IAM dan pengguna root memiliki kredensi jangka panjang. Kredensi pengguna root menyediakan akses penuh ke Akun AWS, sementara pengguna lain memiliki kredensial yang menyediakan akses ke sumber daya yang diberikan oleh kebijakan IAM.

Pengalaman masuk berbeda untuk berbagai jenis AWS Management Console 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 pergi ke halaman mana pun di AWS Management Console, 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.

- Pengguna gabungan yang dikelola di penyedia identitas eksternal yang ditautkan ke Akun AWS login menggunakan portal akses perusahaan khusus. Sumber AWS daya yang tersedia untuk pengguna federasi 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.

AWSAntarmuka Baris Perintah (CLI) dan Kit Pengembangan Perangkat Lunak (SDK)

Pengguna IAM Identity Center dan IAM menggunakan metode yang berbeda untuk mengautentikasi kredensialnya saat mereka mengautentikasi melalui CLI atau antarmuka aplikasi (API) di SDK terkait.

Kredensial 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 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 kredensial yang Anda gunakan untuk masuk ke konsol. Tergantung pada pengalaman Anda, Anda mungkin menemukan CLI menjadi metode yang lebih efisien untuk mengelola 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 kredensial 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 kredensial Pusat Identitas IAM.

Jika Anda menggunakan IAM Identity Center, Anda bisa mendapatkan kredensi jangka pendek dari halaman awal portal AWS akses tempat Anda memilih set izin Anda. Kredensi ini memiliki durasi yang ditentukan dan tidak disegarkan secara otomatis. Jika Anda ingin menggunakan kredensi 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 kredensial Pusat Identitas IAM 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 [Mengonfigurasi 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 kredensi 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](#) kredensial di [AWS Panduan Pengguna Antarmuka Baris Perintah AWS](#)

Untuk mempelajari lebih lanjut tentang mengonfigurasi SDK, lihat [Autentikasi Pusat Identitas IAM](#) di [Panduan Referensi AWSSDK dan Alat](#) dan [Peran IAM Di Mana Saja](#) di [Panduan Referensi SDK dan Alat](#). AWS

Akun AWS ID Anda dan aliasnya

Pengguna IAM di akun masuk menggunakan URL web yang menyertakan alias akun atau ID akun. Jika Anda tidak memiliki URL, halaman AWS login mengharuskan Anda memberikan Akun AWS alias atau ID akun.

Jika Anda tidak tahu ID akun atau alias Anda:

- Periksa riwayat browser Anda. Jika Anda telah masuk sebelumnya, itu dapat disimpan di situs web terbaru Anda.
- Jika Anda telah mengonfigurasi AWS CLI atau AWS SDK dengan kredensi akun Anda, Anda dapat memperoleh ID akun dari file konfigurasi Anda.
- Tanyakan administrator lokal atau pemilik akun Anda, AWS tidak dapat memberikan ID akun kepada pengguna.

Tip

Untuk membuat bookmark untuk halaman masuk akun di browser web, Anda harus mengetikkan URL masuk secara manual di entri bookmark. Jangan gunakan fitur “bookmark halaman ini” browser web Anda karena fitur tersebut menangkap informasi khusus untuk sesi browser Anda saat ini yang mengganggu kunjungan masa depan ke halaman masuk.

Topik

- [Lihat Akun AWS ID Anda](#)
- [Tentang alias akun](#)
- [Membuat, menghapus, dan membuat daftar alias Akun AWS](#)

Lihat Akun AWS ID Anda

Anda dapat melihat ID akun untuk Anda Akun AWS menggunakan metode berikut.

Melihat ID Akun Anda menggunakan konsol

ID akun ditampilkan di dasbor IAM di Akun AWS bagian ini. Ada cara tambahan untuk melihat ID akun Anda di konsol tergantung pada jenis pengguna Anda. Jika Anda telah mengambil peran, kredensi keamanan tidak tersedia.

Jenis pengguna	Prosedur
Pengguna root	Di bilah navigasi di kanan atas, pilih nama pengguna Anda dan kemudian pilih Kredensi keamanan. Nomor akun muncul di bawah Pengidentifikasi akun.
Pengguna IAM	Di bilah navigasi di kanan atas, pilih nama pengguna Anda, ID akun ditampilkan di atas nama pengguna Anda. Pilih Kredensia I keamanan. Nomor akun muncul di bawah Detail akun.
Pengguna federasi	Di bilah navigasi di kanan atas, pilih nama pengguna Anda, ID akun ditampilkan di atas nama pengguna Anda.
Asumsi peran	Di bilah navigasi di kanan atas, pilih ikon Support, lalu pilih Support Center dari daftar. Nomor akun (ID) 12 digit Anda yang masuk saat ini muncul di panel navigasi Pusat Dukungan.

Lihat ID Akun Anda menggunakan AWS CLI

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

- [aws sts get-caller-identity](#)

Melihat ID Akun Anda menggunakan API

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

- [GetCallerIdentity](#)

Tentang alias akun

Jika Anda ingin URL untuk halaman login berisi nama perusahaan Anda (atau pengenalan ramah lainnya) alih-alih Akun AWS ID Anda, Anda dapat membuat alias akun. Bagian ini memberikan informasi tentang Akun AWS alias dan mencantumkan operasi API yang Anda gunakan untuk membuat alias.

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

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

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

- Anda hanya Akun AWS dapat memiliki satu alias. Jika membuat alias baru untuk akun AWS Anda, alias baru tersebut akan menimpa alias sebelumnya, dan URL yang berisi alias sebelumnya berhenti bekerja.
- Alias akun harus berisi hanya digit, huruf kecil, dan tanda hubung. Untuk informasi selengkapnya tentang batasan entitas AWS akun, lihat [IAM dan AWS STS kuota](#).
- Alias akun harus unik di semua produk Amazon Web Services dalam partisi jaringan tertentu.

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

Berikut ini adalah partisi yang didukung:

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

Membuat, menghapus, dan membuat daftar alias Akun AWS

Anda dapat menggunakan AWS Management Console, API IAM, atau antarmuka baris perintah untuk membuat atau menghapus Akun AWS alias Anda.

Note

Alias akun bukanlah rahasia, dan mereka akan muncul di URL halaman masuk Anda yang menghadap publik. 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 Anda.

Membuat atau mengedit alias akun (konsol)

Anda dapat membuat, mengedit, dan menghapus alias akun dari AWS Management Console.

Izin minimum

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

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

Untuk membuat atau mengedit alias akun (konsol)

1. Masuk ke AWS Management Console 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.

Note

Anda hanya dapat memiliki satu alias yang terkait dengan Akun AWS pada satu waktu. Jika Anda membuat alias baru, alias sebelumnya akan dihapus, dan URL login yang dikaitkan dengan alias sebelumnya berhenti berfungsi.

Hapus alias akun (konsol)

Anda dapat menghapus alias akun dari AWS Management Console

Izin minimum

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

- `iam:ListAccountAliases`
- `iam:CreateAccountAlias`
- `iam>DeleteAccountAlias`

Untuk menghapus alias akun (konsol)

1. Masuk ke AWS Management Console 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.

Note

Satu-satunya URL masuk untuk akun Anda didasarkan pada ID akun Anda. Setiap upaya untuk terhubung ke URL alias tidak dialihkan.

Membuat, menghapus, dan mencantumkan alias (AWS CLI)

Note

Untuk menggunakan perintah berikut, Anda harus memiliki setidaknya izin IAM berikut:

- `iam:ListAccountAliases`
- `iam:CreateAccountAlias`
- `iam>DeleteAccountAlias`

Untuk membuat alias untuk URL halaman AWS Management Console masuk Anda, jalankan perintah berikut:

- [`aws iam create-account-alias`](#)

Untuk menghapus alias Akun AWS ID, jalankan perintah berikut:

- [`aws iam delete-account-alias`](#)

Untuk menampilkan alias Akun AWS ID Anda, jalankan perintah berikut:

- [`aws iam list-account-aliases`](#)

Example Perintah alias

Untuk menampilkan alias Akun AWS ID Anda, jalankan perintah berikut.

```
$ aws iam list-account-aliases
{
  "AccountAliases": [
    "myaccountalias"
  ]
}
```

Untuk membuat alias untuk AWS Management Console login Anda, jalankan perintah berikut:

```
$ aws iam create-account-alias \
```

```
--account-alias myaliasname
```

Perintah ini tidak menghasilkan output jika berhasil.

Untuk menghapus alias Akun AWS ID, jalankan perintah berikut.

```
$ aws iam delete-account-alias \  
--account-alias myaliasname
```

Perintah ini tidak menghasilkan output jika berhasil.

Membuat, menghapus, dan mendaftarkan alias (API AWS)

Note

Untuk menggunakan operasi API berikut, Anda harus memiliki setidaknya izin IAM berikut:

- `iam:ListAccountAliases`
- `iam:CreateAccountAlias`
- `iam>DeleteAccountAlias`

Untuk membuat alias untuk URL halaman AWS Management Console masuk Anda, panggil operasi berikut:

- [CreateAccountAlias](#)

Untuk menghapus alias Akun AWS ID, panggil operasi berikut:

- [DeleteAccountAlias](#)

Untuk menampilkan alias Akun AWS ID Anda, hubungi operasi berikut:

- [ListAccountAliases](#)

Memulai dengan IAM

Gunakan tutorial ini untuk memulai AWS Identity and Access Management (IAM). Anda akan mempelajari cara membuat peran, pengguna, dan kebijakan menggunakan AWS Management Console.

AWS Identity and Access Management adalah fitur dari Akun AWS yang ditawarkan tanpa biaya tambahan. Anda akan dikenakan biaya hanya untuk penggunaan AWS produk lain oleh pengguna IAM Anda. Untuk informasi tentang harga produk AWS lainnya, lihat [Halaman harga Amazon Web Services](#).

Note

Rangkaian dokumentasi ini utamanya berkaitan dengan layanan IAM. Untuk mempelajari tentang memulai dengan AWS dan menggunakan beberapa layanan untuk memecahkan masalah seperti membangun dan meluncurkan proyek pertama Anda, lihat [Memulai Pusat Sumber Daya](#).

Daftar Isi

- [Prasyarat](#)
- [Buat pengguna IAM pertama Anda](#)
- [Buat peran pertama Anda](#)
- [Buat kebijakan IAM pertama Anda](#)
- [Akses terprogram](#)

Prasyarat

Sebelum memulai, pastikan Anda telah menyelesaikan langkah-langkah tersebut di [Menyiapkan IAM](#). Tutorial ini menggunakan akun administrator yang Anda buat dalam prosedur itu.

Buat pengguna IAM pertama Anda

[Pengguna IAM](#) adalah identitas dalam Akun AWS Anda yang memiliki izin khusus untuk satu orang atau aplikasi. Pengguna dapat diatur ke dalam grup yang berbagi izin yang sama.

Note

Sebagai [praktik keamanan terbaik](#), kami menyarankan Anda menyediakan akses ke sumber daya Anda 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 tujuan membiasakan diri dengan proses pembuatan pengguna IAM, tutorial ini memberi Anda langkah untuk membuat pengguna dan grup IAM untuk akses darurat.

Untuk membuat pengguna IAM pertama Anda

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, pilih layanan IAM.
3. Di panel navigasi, pilih Pengguna dan kemudian pilih Tambah pengguna.

Note

Jika Anda mengaktifkan Pusat Identitas IAM, akan AWS Management Console menampilkan pengingat bahwa yang terbaik adalah mengelola akses pengguna di Pusat Identitas IAM. Dalam tutorial ini, pengguna IAM yang Anda buat khusus untuk digunakan hanya ketika pengguna Anda di kredensi IAM Identity Center tidak tersedia.

4. Untuk Nama pengguna, masukkan **EmergencyAccess**. Nama tidak dapat berisi spasi.
5. Pilih kotak centang di sebelah Berikan akses pengguna ke AWS Management Console — opsional dan kemudian pilih Saya ingin membuat pengguna IAM.
6. Di bawah Kata sandi konsol, pilih Kata sandi yang dibuat otomatis.
7. Kosongkan kotak centang di sebelah Pengguna harus membuat kata sandi baru saat masuk berikutnya (disarankan). Karena pengguna IAM ini untuk akses darurat, administrator tepercaya mempertahankan kata sandi dan hanya menyediakannya saat diperlukan.
8. Pada halaman Setel 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 izin, pilih **AdministratorAccess**.
10. Pilih Buat grup pengguna untuk kembali ke halaman Setel izin.
11. Di bawah Grup pengguna, pilih nama yang **EmergencyAccessGroup** Anda buat sebelumnya.
12. Pilih Berikutnya untuk melanjutkan ke halaman Tinjauan dan buat.
13. Pada halaman Tinjau 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 federasi Anda.

Pengguna IAM baru ditampilkan dalam daftar Pengguna. Pilih tautan Nama pengguna untuk melihat detail pengguna. Di bawah Ringkasan, salin ARN pengguna ke clipboard. Tempelkan ARN ke dalam dokumen teks, sehingga Anda dapat menggunakannya dalam prosedur selanjutnya.

Buat peran pertama Anda

Peran IAM adalah cara aman untuk memberikan izin kepada entitas yang Anda percayai. Peran IAM memiliki beberapa kesamaan dengan pengguna IAM. Peran dan pengguna keduanya merupakan prinsip 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. Menggunakan peran membantu Anda mengikuti praktik terbaik IAM. Anda dapat menggunakan peran untuk:

- Aktifkan identitas tenaga kerja dan akses aplikasi yang diaktifkan Pusat Identitas ke penggunaan. AWS Management Console AWS IAM Identity Center
- Delegasikan izin ke AWS layanan untuk melakukan tindakan atas nama Anda.
- Aktifkan kode aplikasi yang berjalan pada instans Amazon EC2 untuk mengakses atau memodifikasi AWS sumber daya.
- Berikan akses ke yang lain Akun AWS.

 Note

Anda dapat menggunakan AWS Identity and Access Management Peran Di Mana Saja untuk memberikan akses ke identitas mesin. Menggunakan Peran IAM Di Mana Saja berarti Anda tidak perlu mengelola kredensial jangka panjang untuk beban kerja yang berjalan di luar. AWS Untuk informasi selengkapnya, lihat [Apa itu AWS Identity and Access Management Peran Di Mana Saja?](#) di Panduan Pengguna AWS Identity and Access Management Peran Di Mana Saja.

IAM Identity Center dan AWS layanan lainnya secara otomatis membuat peran untuk layanan mereka. Jika Anda menggunakan pengguna IAM, kami sarankan Anda membuat peran untuk diambil oleh pengguna Anda saat mereka masuk. Ini akan memberi mereka izin sementara selama sesi, bukan izin jangka panjang.

AWS Management Console Wizard yang memandu Anda melalui langkah-langkah untuk membuat peran menampilkan langkah-langkah yang sedikit berbeda tergantung pada apakah Anda membuat peran untuk pengguna IAM, AWS layanan, atau untuk pengguna federasi. Akses reguler ke Akun AWS dalam organisasi harus disediakan menggunakan akses federasi. Jika Anda membuat pengguna IAM untuk tujuan tertentu, seperti akses darurat atau akses terprogram, hanya berikan izin kepada pengguna IAM tersebut untuk mengambil peran dan menempatkan pengguna IAM tersebut ke dalam grup khusus peran.

Dalam prosedur ini, Anda membuat peran yang menyediakan SupportUser akses untuk pengguna EmergencyAccess IAM. Sebelum memulai prosedur ini, salin ARN pengguna IAM ke clipboard.

Untuk membuat peran bagi pengguna IAM

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, pilih layanan IAM.
3. Di panel navigasi konsol IAM, pilih Peran kemudian pilih Buat peran.
4. Pilih tipe Akun AWS peran.
5. Di Pilih entitas tepercaya, di bawah Jenis entitas tepercaya, pilih Kebijakan kepercayaan khusus.
6. Di bagian Kebijakan kepercayaan khusus, tinjau kebijakan kepercayaan dasar. Inilah yang akan kita gunakan untuk peran ini. Gunakan editor pernyataan Edit untuk memperbarui kebijakan kepercayaan:

1. Di Tambahkan tindakan untuk STS, pilih Asumsikan Peran.
2. Di samping Tambahkan prinsipal, pilih, Tambah. Jendela Add principal terbuka.

Di bawah Jenis utama, pilih Pengguna IAM.

Di bawah ARN, tempel ARN pengguna IAM yang Anda salin ke clipboard.

Pilih Tambahkan prinsipal.

3. Verifikasi bahwa Principal baris dalam kebijakan kepercayaan sekarang berisi ARN yang Anda tentukan:

```
"Principal": { "AWS": "arn:aws:iam::123456789012:user/username" }
```

7. Selesaikan peringatan keamanan, kesalahan, atau peringatan umum yang dihasilkan selama [validasi kebijakan](#), lalu pilih Berikutnya.
8. Di Tambahkan izin, pilih kotak centang di samping kebijakan izin yang akan diterapkan. Untuk tutorial ini kita akan memilih kebijakan SupportUserkepercayaan. Anda kemudian dapat menggunakan peran ini untuk memecahkan masalah dan menyelesaikan masalah dengan Akun AWS dan membuka kasus dukungan dengan. AWS Kami tidak akan menetapkan [batas izin](#) saat ini.
9. Pilih Berikutnya.
10. Di Nama, tinjau, dan buat setelan lengkap ini:
 - Untuk nama Peran, masukkan nama yang mengidentifikasi peran ini, seperti SupportUserRole.
 - Untuk Deskripsi, jelaskan tujuan penggunaan peran tersebut.

Karena sumber daya AWS lainnya mungkin merujuk peran tersebut, Anda tidak dapat mengubah nama peran tersebut setelah nama dibuat.

11. Pilih Buat peran.

Setelah peran dibuat, bagikan informasi peran dengan orang-orang yang membutuhkan peran tersebut. Anda dapat membagikan informasi peran dengan:

- Tautan peran: Kirim pengguna tautan 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.
- Menyimpan informasi tautan peran bersama dengan kredensi EmergencyAccess pengguna.

Untuk detailnya, lihat [Memberikan informasi kepada pengguna](#).

Buat kebijakan IAM pertama Anda

Kebijakan IAM dilampirkan pada identitas IAM (pengguna, kelompok pengguna, atau peran) atau sumber daya. AWS Kebijakan adalah objek di AWS, yang saat terkait dengan identitas atau sumber daya, akan menentukan izinnya.

Untuk membuat kebijakan IAM pertama Anda

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, pilih layanan IAM.
3. Di panel navigasi, pilih Kebijakan.

Jika ini pertama kalinya Anda memilih Kebijakan, akan muncul laman Selamat Datang di Kebijakan Terkelola. Pilih Memulai.

4. Pilih Buat kebijakan.
5. Pada halaman Buat kebijakan, pilih Tindakan, lalu pilih Kebijakan impor.
6. Di jendela Kebijakan impor, di kotak Temukan kebijakan, ketik **power** untuk mengurangi daftar kebijakan. Pilih PowerUserAccesskebijakan.
7. Pilih Kebijakan impor. Kebijakan ditampilkan di tab JSON.
8. Pilih Berikutnya.
9. Pada halaman Tinjau dan buat, untuk nama Kebijakan, ketik **PowerUserExamplePolicy**. Untuk Deskripsi, ketik **Allows full access to all services except those for user management**. Kemudian pilih Buat kebijakan untuk menyimpan kebijakan.

Anda dapat melampirkan kebijakan ini ke peran untuk memberikan izin yang terkait dengan kebijakan ini kepada pengguna yang menganggap peran tersebut. `PowerUserAccessKebijakan` ini umumnya digunakan untuk menyediakan akses ke pengembang.

Akses terprogram

Pengguna membutuhkan akses terprogram jika mereka ingin berinteraksi dengan AWS di luar AWS Management Console. 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 waktu kedaluwarsa kredensial.

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

Pengguna mana yang membutuhkan akses terprogram?	Ke	Oleh
Identitas tenaga kerja (Pengguna yang dikelola di Pusat Identitas IAM)	Gunakan kredensial jangka pendek untuk menandatangani permintaan terprogram ke AWS CLI atau API AWS (secara langsung atau menggunakan SDK AWS).	Mengikuti instruksi untuk antarmuka yang ingin Anda gunakan: <ul style="list-style-type: none"> • Untuk itu AWS CLI, ikuti petunjuk dalam Mendapatkan kredensi peran IAM untuk akses CLI di Panduan Pengguna. AWS IAM Identity Center • Untuk AWS API, ikuti petunjuk dalam kredensial SSO di AWSSDK dan Panduan Referensi Alat.

Pengguna mana yang membutuhkan akses terprogram?	Ke	Oleh
IAM	Gunakan kredensial jangka pendek untuk menandatangani permintaan terprogram ke AWS CLI atau API AWS (secara langsung atau menggunakan SDK AWS).	Mengikuti instruksi dalam Menggunakan kredensial sementara dengan AWS sumber daya.
IAM	Gunakan kredensial jangka panjang untuk menandatangani permintaan terprogram ke AWS CLI atau API AWS (secara langsung atau menggunakan SDK AWS). (Tidak disarankan)	Mengikuti petunjuk dalam Mengelola kunci akses untuk pengguna IAM .

Praktik terbaik keamanan dan kasus penggunaan AWS Identity and Access Management

AWS Identity and Access Management(IAM) menyediakan sejumlah fitur keamanan untuk dipertimbangkan ketika Anda mengembangkan dan menerapkan kebijakan keamanan Anda sendiri. Praktik terbaik berikut adalah pedoman umum dan tidak mewakili solusi keamanan yang lengkap. Karena praktik terbaik ini mungkin tidak sesuai atau cukup untuk lingkungan Anda, anggap sebagai pertimbangan yang membantu dan bukan sebagai resep.

Untuk mendapatkan manfaat terbesar dari IAM, luangkan waktu untuk mempelajari praktik terbaik yang direkomendasikan. Salah satu cara untuk melakukannya adalah untuk melihat bagaimana IAM digunakan dalam skenario dunia nyata untuk bekerja dengan layanan lain 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](#)

Praktik AWS Identity and Access Management terbaik diperbarui pada 14 Juli 2022.

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

Topik

- [Mewajibkan pengguna manusia untuk menggunakan federasi dengan penyedia identitas untuk mengakses AWS menggunakan kredensi sementara](#)
- [Memerlukan beban kerja untuk menggunakan kredensial sementara dengan peran IAM untuk mengakses AWS](#)
- [Membutuhkan otentikasi multi-faktor \(MFA\)](#)

- [Perbarui kunci akses bila diperlukan untuk kasus penggunaan yang memerlukan kredensi jangka panjang](#)
- [Ikuti praktik terbaik untuk melindungi kredensial pengguna root Anda](#)
- [Menerapkan izin hak istimewa paling sedikit](#)
- [Memulai kebijakan AWS terkelola dan beralih ke izin hak istimewa paling sedikit](#)
- [Gunakan IAM Access Analyzer untuk menghasilkan kebijakan hak istimewa paling sedikit 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](#)
- [Memverifikasi akses publik dan lintas akun dengan IAM Access Analyzer](#)
- [Gunakan IAM Access Analyzer untuk memvalidasi kebijakan IAM Anda guna memastikan izin yang aman dan fungsional](#)
- [Menetapkan pagar pembatas izin di beberapa akun](#)
- [Gunakan batas izin untuk mendelegasikan manajemen izin dalam akun](#)

Mewajibkan pengguna manusia untuk menggunakan federasi dengan penyedia identitas untuk mengakses AWS menggunakan kredensi 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 menjadi 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.

Minta pengguna manusia Anda untuk menggunakan kredensi sementara saat mengakses. AWS Anda dapat menggunakan penyedia identitas bagi pengguna manusia Anda untuk menyediakan akses federasi Akun AWS dengan mengambil peran, yang menyediakan kredensi sementara. Untuk manajemen akses terpusat, kami sarankan Anda menggunakan [AWS IAM Identity Center \(IAM Identity Center\)](#) untuk mengelola akses ke akun Anda dan izin 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.

Untuk informasi lebih lanjut tentang peran, lihat [Istilah dan konsep peran](#).

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

Beban 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 identitas untuk membuat permintaan Layanan AWS, seperti permintaan untuk membaca data. Identitas ini mencakup mesin yang berjalan di AWS lingkungan Anda, seperti instans AWS Lambda atau 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 kredensial 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 [AWS Identity and Access Management Peran Di Mana Saja](#). Untuk informasi lebih lanjut tentang peran, lihat [Peran IAM](#). Untuk informasi lebih lanjut tentang cara menggunakan peran untuk mendelegasikan akses Akun AWS, lihat [Tutorial IAM: Delegasikan akses AWS Akun yang menggunakan peran IAM](#).

Mebutuhkan 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. Setiap kredensial pengguna dan respons yang dihasilkan perangkat diperlukan untuk menyelesaikan proses login. Untuk informasi selengkapnya, lihat [Menggunakan otentikasi multi-faktor \(MFA\) di AWS](#).

Jika Anda menggunakan Pusat Identitas IAM untuk manajemen akses terpusat bagi pengguna manusia, Anda dapat menggunakan kemampuan MFA Pusat Identitas IAM ketika sumber identitas Anda dikonfigurasi dengan penyimpanan identitas Pusat Identitas IAM, AWS Microsoft AD Terkelola, atau Konektor AD. Untuk informasi selengkapnya tentang MFA di Pusat Identitas IAM, lihat [Autentikasi multi-faktor](#) di Panduan Pengguna. AWS IAM Identity Center

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

Jika memungkinkan, sebaiknya mengandalkan kredensi sementara alih-alih membuat kredensial jangka panjang seperti kunci akses. Namun, untuk skenario di mana Anda memerlukan pengguna IAM dengan akses terprogram dan kredensi jangka panjang, kami sarankan Anda memperbarui kunci akses saat diperlukan, seperti ketika seorang karyawan meninggalkan perusahaan Anda. Kami menyarankan Anda menggunakan akses IAM informasi yang terakhir digunakan untuk memperbarui dan menghapus kunci akses dengan aman. Untuk informasi selengkapnya, lihat [Memperbarui kunci akses](#).

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

- **Beban kerja yang tidak dapat menggunakan peran IAM** — Anda dapat menjalankan beban kerja dari lokasi yang perlu diakses. AWS Dalam beberapa situasi, Anda tidak dapat menggunakan peran IAM untuk memberikan kredensi sementara, seperti untuk plugin. WordPress Dalam situasi ini, gunakan kunci akses jangka panjang pengguna IAM agar beban kerja tersebut dapat diautentikasi. AWS
- **AWSKlien pihak ketiga** — Jika Anda menggunakan alat yang tidak mendukung akses dengan IAM Identity Center, seperti AWS klien pihak ketiga atau vendor yang tidak di-hostAWS, gunakan kunci akses jangka panjang pengguna IAM.
- **AWS CodeCommitakses** - Jika Anda menggunakan CodeCommit untuk menyimpan kode Anda, Anda dapat menggunakan pengguna IAM dengan kunci SSH atau kredensi khusus layanan untuk CodeCommit mengautentikasi ke repositori Anda. Kami menyarankan Anda melakukan ini selain menggunakan pengguna di IAM Identity Center untuk otentikasi normal. Pengguna di IAM Identity Center adalah orang-orang di tenaga kerja Anda yang membutuhkan akses ke aplikasi cloud Anda Akun AWS atau ke aplikasi cloud Anda. Untuk memberi pengguna akses ke CodeCommit repositori Anda tanpa mengkonfigurasi pengguna IAM, Anda dapat mengkonfigurasi utilitas. git-remote-codecommit Untuk informasi lebih lanjut tentang IAM dan CodeCommit, lihat [Menggunakan IAM dengan CodeCommit: Kredensial Git, kunci SSH, dan kunci akses AWS](#). Untuk informasi selengkapnya tentang mengonfigurasi git-remote-codecommit utilitas, lihat [Menghubungkan ke AWS CodeCommit repositori dengan kredensi berputar](#) di Panduan Pengguna. AWS CodeCommit
- **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 kredensi khusus layanan untuk

mengautentikasi dengan Amazon Keyspaces. Pengguna di IAM Identity Center adalah orang-orang di tenaga kerja Anda yang membutuhkan akses ke aplikasi cloud Anda Akun AWS atau ke aplikasi cloud Anda. Anda juga dapat terhubung ke Amazon Keyspaces menggunakan kredensial sementara. Untuk informasi lebih lanjut, lihat [Mengakses Amazon Keyspaces menggunakan peran IAM dan plugin SiGv4 dalam 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 file. AWS Management Console Lindungi kredensial 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, lihat [Praktik terbaik pengguna root untuk Anda Akun AWS](#).

Menerapkan izin hak istimewa paling sedikit

Saat Anda menetapkan izin dengan kebijakan IAM, berikan hanya izin yang diperlukan untuk melakukan tugas. Anda melakukan ini dengan mendefinisikan tindakan yang dapat diambil pada sumber daya tertentu dalam kondisi tertentu, juga dikenal sebagai izin hak istimewa paling sedikit. Anda dapat memulai dengan izin luas saat 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 lebih lanjut tentang penggunaan IAM untuk menerapkan izin, lihat [Kebijakan dan Izin di IAM](#)

Memulai kebijakan AWS terkelola dan beralih ke izin hak istimewa paling sedikit

Untuk mulai memberikan izin kepada pengguna dan beban kerja Anda, gunakan kebijakan AWS terkelola yang memberikan izin untuk banyak kasus penggunaan umum. Mereka tersedia di Akun AWS. 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 yang dikelola](#). Untuk informasi selengkapnya tentang kebijakan AWS terkelola yang dirancang untuk fungsi pekerjaan tertentu, lihat [AWS kebijakan terkelola untuk fungsi pekerjaan](#).

Gunakan IAM Access Analyzer untuk menghasilkan kebijakan hak istimewa paling sedikit 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, lalu 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](#).

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

Anda mungkin memiliki pengguna IAM, peran, izin, kebijakan, atau kredensial yang tidak lagi Anda perlukan. Akun AWS IAM menyediakan informasi yang terakhir diakses untuk membantu Anda mengidentifikasi pengguna, peran, izin, kebijakan, dan kredensial 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 Anda agar lebih mematuhi izin hak istimewa paling sedikit. Untuk informasi selengkapnya, lihat [Menyaring izin di 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 SSL. Anda juga dapat menggunakan ketentuan untuk memberikan akses ke tindakan layanan, tetapi hanya jika digunakan melalui yang spesifik Layanan AWS, seperti AWS CloudFormation. Untuk informasi selengkapnya, lihat [Elemen kebijakan IAM JSON: Condition](#).

Memverifikasi akses publik dan lintas akun dengan IAM Access Analyzer

Sebelum Anda memberikan izin untuk akses publik atau lintas akun AWS, kami sarankan Anda memverifikasi 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 IAM Access Analyzer. Temuan ini membantu Anda memverifikasi bahwa kontrol akses sumber daya Anda 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 [Mempratinjau akses dengan IAM Access Analyzer API](#).

Gunakan IAM Access Analyzer untuk memvalidasi kebijakan IAM Anda guna memastikan 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 memperbaiki dan memvalidasi kebijakan sebelum menyimpannya. Selain itu, kami menyarankan Anda untuk 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 [IAM Access Analyzer](#).

Menetapkan pagar pembatas izin di beberapa akun

Saat Anda menskalakan beban kerja Anda, pisahkan dengan menggunakan beberapa akun yang dikelola. AWS Organizations Kami menyarankan Anda menggunakan [kebijakan kontrol layanan](#) Organizations (SCP) untuk membuat pagar pembatas izin untuk mengontrol akses bagi semua pengguna dan peran IAM di seluruh akun Anda. SCP adalah jenis kebijakan organisasi yang dapat Anda gunakan untuk mengelola izin dalam organisasi Anda di tingkat AWS organisasi, OU, atau akun. Pagar izin yang Anda buat berlaku untuk semua pengguna dan peran dalam akun yang tercakup. Namun, SCP saja tidak cukup untuk memberikan izin ke akun dalam organisasi Anda.

Untuk melakukan ini, administrator Anda harus melampirkan [kebijakan berbasis identitas atau sumber daya](#) untuk pengguna IAM, peran IAM, atau sumber daya dalam akun Anda. Untuk informasi lebih lanjut, lihat [AWS Organizations, akun, dan pagar pembatas IAM](#).

Gunakan batas izin untuk mendelegasikan manajemen izin dalam akun

Dalam beberapa skenario, Anda mungkin ingin mendelegasikan manajemen 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 oleh kebijakan berbasis identitas untuk peran IAM. Batas izin tidak memberikan izin sendiri. Untuk informasi selengkapnya, lihat [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 kredensi pengguna root](#). Anda perlu mengamankan kredensi pengguna root dan mekanisme pemulihan akun Anda untuk membantu memastikan Anda tidak mengekspos kredensial Anda yang sangat istimewa untuk penggunaan yang tidak sah.

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

- Untuk satu, mandiri Akun AWS, lihat [Membuat pengguna administratif](#).
- Untuk beberapa 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 Akun AWS. Kami sangat menyarankan Anda meminta pengguna untuk mengautentikasi 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 diasumsikan oleh siapa saja yang membutuhkannya. Juga, 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 kredensi jangka panjang seperti kata sandi dan kunci akses. Jika memungkinkan, [praktik terbaik](#) merekomendasikan untuk mengandalkan kredensial sementara alih-alih membuat pengguna IAM yang memiliki kredensi jangka panjang seperti kata sandi dan kunci akses.

- Untuk beberapa yang Akun AWS dikelola melalui Organizations, gunakan pengguna tenaga kerja IAM Identity Center. Dengan IAM Identity Center, Anda dapat mengelola pengguna secara terpusat di seluruh akun Anda Akun AWS dan izin di dalam akun tersebut. Kelola identitas pengguna Anda dengan IAM Identity Center atau dari penyedia identitas eksternal. Untuk informasi selengkapnya, lihat [Apa itu AWS IAM Identity Center](#) dalam Panduan Pengguna AWS IAM Identity Center .

Topik

- [Amankan kredensi 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 login pengguna root sedapat mungkin](#)
- [Menggunakan alamat email grup untuk kredensi pengguna root](#)
- [Batasi akses ke mekanisme pemulihan akun](#)
- [Amankan kredensi pengguna root akun Organizations Anda](#)
- [Memantau akses dan penggunaan](#)

Amankan kredensi pengguna root Anda untuk mencegah penggunaan yang tidak sah

Amankan kredensi pengguna root Anda dan gunakan hanya untuk [tugas-tugas yang membutuhkannya](#). Untuk membantu mencegah penggunaan yang tidak sah, jangan bagikan kata sandi pengguna root, MFA, kunci akses, pasangan kunci CloudFront , 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. Kami menyarankan 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 algoritme pembuatan kata sandi yang kuat dapat membantu Anda mencapai tujuan ini. AWS mengharuskan kata sandi Anda memenuhi ketentuan berikut:

- Itu harus memiliki minimal 8 karakter dan maksimal 128 karakter.
- Ini harus mencakup minimal tiga dari campuran tipe 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 kredensi masuk. Kami sangat menyarankan untuk mengaktifkan beberapa MFA untuk kredensial pengguna root Anda untuk memberikan fleksibilitas dan ketahanan tambahan dalam strategi keamanan Anda. Anda dapat mendaftarkan hingga delapan perangkat MFA dari kombinasi jenis MFA yang saat ini didukung dengan pengguna root Anda. Akun AWS

- Kunci keamanan perangkat keras Bersertifikat FIDO disediakan oleh penyedia pihak ketiga. Untuk informasi selengkapnya, lihat [Mengaktifkan 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 [Mengaktifkan token TOTP perangkat keras untuk pengguna Akun AWS root](#).
- Aplikasi otentikator virtual yang berjalan di ponsel atau perangkat lain dan mengemulasi perangkat fisik. Untuk informasi selengkapnya, lihat [Mengaktifkan perangkat MFA virtual untuk pengguna Akun AWS root Anda](#).

Jangan membuat kunci akses untuk pengguna root

Kunci akses memungkinkan Anda menjalankan perintah di AWS Command Line Interface (AWS CLI) atau menggunakan operasi API dari salah satu AWS SDK. Kami sangat menyarankan agar Anda tidak membuat pasangan 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, kami sarankan masuk ke AWS Management Console untuk melakukan tugas pengguna root. Sebelum membuat kunci akses, tinjau [alternatif untuk kunci akses jangka panjang](#).

Gunakan persetujuan multi-orang untuk login pengguna root sedapat mungkin

Pertimbangkan untuk menggunakan persetujuan multi-orang untuk memastikan bahwa tidak ada satu orang pun yang dapat mengakses MFA dan kata sandi untuk pengguna root. Beberapa perusahaan menambahkan lapisan keamanan tambahan dengan menyiapkan satu grup administrator dengan akses ke kata sandi, dan grup administrator lain dengan akses ke MFA. Satu anggota dari setiap grup harus berkumpul untuk masuk sebagai pengguna root.

Menggunakan alamat email grup untuk kredensi pengguna root

Gunakan alamat email yang dikelola oleh bisnis Anda dan teruskan pesan yang diterima langsung ke sekelompok pengguna. Jika AWS harus menghubungi pemilik akun, pendekatan ini mengurangi risiko keterlambatan dalam menanggapi, 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 kredensial pengguna root yang sama. Jika

Anda kehilangan akses ke semua LKM Anda, Anda memerlukan nomor telepon dan email yang digunakan untuk mendaftarkan akun Anda, agar mutakhir 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 Anda, maka nomor telepon yang terdaftar di akun Anda dapat digunakan sebagai cara alternatif untuk memulihkan kredensi pengguna root. Pastikan Anda memiliki akses ke nomor telepon kontak, perbarui nomor telepon, dan batasi siapa yang memiliki akses untuk mengelola nomor telepon.

Tidak seorang pun harus 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 yang memiliki akses ke nomor telepon utama untuk memulihkan akses ke akun Anda sebagai pengguna root.

Amankan kredensi pengguna root akun Organizations Anda

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

Amankan kredensi pengguna root untuk akun anggota

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

- Amankan kredensi pengguna root dari akun Organizations Anda dengan MFA.
- Jangan mengatur ulang kata sandi pengguna root untuk akun Anda, dan hanya memulihkan akses ke sana bila diperlukan menggunakan proses reset kata sandi. Saat Anda membuat akun anggota di organisasi Anda, Organizations secara otomatis membuat peran IAM di akun anggota yang memungkinkan akun manajemen akses sementara ke akun anggota.

Untuk detailnya, lihat [Mengakses akun anggota di organisasi Anda](#) di Panduan Pengguna Organizations.

Menetapkan kontrol keamanan preventif di Organizations menggunakan kebijakan kontrol layanan (SCP)

Jika Anda menggunakan Organizations untuk mengelola beberapa akun, Anda dapat menerapkan SCP untuk membatasi akses ke pengguna root akun anggota. Menyangkal 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

Kami menyarankan Anda menggunakan mekanisme pelacakan saat ini untuk memantau, memperingatkan, dan melaporkan login dan penggunaan kredensial pengguna root, termasuk peringatan yang mengumumkan login dan penggunaan pengguna root. Layanan berikut dapat membantu memastikan bahwa penggunaan kredensial pengguna root dilacak dan melakukan pemeriksaan keamanan yang dapat membantu mencegah penggunaan yang tidak sah.

- Jika Anda ingin diberi tahu tentang aktivitas login pengguna root di akun Anda, Anda dapat memanfaatkan Amazon CloudWatch untuk membuat aturan Acara yang mendeteksi kapan kredensial pengguna root digunakan dan memicu pemberitahuan ke administrator keamanan Anda. Untuk detailnya, lihat [Memantau dan memberi tahu aktivitas pengguna Akun AWS root](#).
- Jika Anda ingin mengatur 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 [Mengirim pemberitahuan saat objek Amazon S3 dibuat](#).
- Jika Anda sudah menggunakan GuardDuty sebagai layanan deteksi ancaman, Anda dapat [memperluas kemampuannya](#) untuk memberi tahu Anda ketika kredensial pengguna root digunakan di akun Anda.

Peringatan harus mencakup, tetapi tidak terbatas pada, alamat email untuk pengguna root. Memiliki prosedur untuk menanggapi peringatan sehingga personel yang menerima peringatan akses pengguna root memahami cara memvalidasi bahwa akses pengguna root diharapkan, dan bagaimana meningkatkannya jika mereka yakin bahwa insiden keamanan sedang berlangsung. Untuk contoh cara mengonfigurasi peringatan, lihat [Memantau dan memberi tahu aktivitas pengguna Akun AWS root](#).

Evaluasi kepatuhan MFA pengguna root

- AWS Config menggunakan aturan untuk membantu menegakkan praktik terbaik pengguna root. Anda dapat menggunakan aturan AWS terkelola untuk [mengharuskan pengguna root mengaktifkan autentikasi multi-faktor \(MFA\)](#). AWS Config juga dapat [mengidentifikasi kunci akses untuk pengguna root](#).
- Security Hub 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 Security Hub.
- Trusted Advisor menyediakan pemeriksaan keamanan sehingga Anda tahu apakah MFA tidak diaktifkan pada akun pengguna root. Untuk informasi selengkapnya, lihat [MFA on Root Account](#) di AWS Support User Guide.

Jika Anda perlu melaporkan masalah keamanan di akun, lihat [Melaporkan Email Mencurigakan](#) atau Pelaporan [Kerentanan](#). Atau, Anda dapat [menghubungi AWS](#) untuk bantuan dan panduan tambahan.

Kasus penggunaan bisnis untuk IAM

Kasus penggunaan bisnis sederhana untuk IAM dapat membantu Anda memahami cara-cara dasar Anda dapat mengimplementasikan 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 lebih lanjut tentang penggunaan IAM dengan layanan lain dari AWS, lihat [AWS 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 Contoh Corp Setelah memulai perusahaan, mereka membuat Akun AWS dan mendirikan AWS IAM Identity Center (IAM Identity Center) untuk membuat akun administratif untuk digunakan dengan AWS sumber daya mereka. Saat Anda menyiapkan akses akun untuk pengguna administratif, IAM Identity Center akan membuat peran IAM yang sesuai. Peran ini, yang dikendalikan oleh IAM Identity Center, dibuat dalam yang relevan Akun AWS, dan kebijakan yang ditentukan dalam kumpulan AdministratorAccessizin 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 keamanan terbaik, mereka mengkonfigurasi Multi-Factor Authentication (MFA) untuk kredensial pengguna root mereka dan memutuskan cara melindungi kredensial pengguna root mereka.

Seiring pertumbuhan perusahaan mereka, mereka mempekerjakan karyawan untuk bekerja sebagai pengembang, admin, pengujian, 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 memberikan karyawan mereka akses ke AWS sumber daya, mereka menggunakan IAM Identity Center untuk menghubungkan Active Directory perusahaan mereka untuk mereka Akun AWS.

Karena mereka menghubungkan Active Directory ke IAM Identity Center, pengguna, grup, dan keanggotaan grup disinkronkan dan didefinisikan. Mereka harus menetapkan set izin dan peran untuk kelompok yang berbeda untuk memberikan pengguna tingkat yang benar akses ke AWS sumber daya. Mereka menggunakan [AWS kebijakan terkelola untuk fungsi pekerjaan](#) dalam AWS Management Console untuk membuat set izin ini:

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

Kemudian mereka menetapkan set izin ini ke peran yang ditetapkan ke grup Active Directory mereka.

Untuk step-by-step panduan yang menjelaskan konfigurasi awal IAM Identity Center, lihat [Memulai](#) di PanduanAWS IAM Identity Center Pengguna. Untuk informasi selengkapnya tentang penyediaan akses pengguna IAM Identity Center, lihat [Akses masuk tunggal keAWS akun](#) di PanduanAWS IAM Identity Center Pengguna.

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 lebih lanjut tentang Amazon EC2, masuk ke [Panduan Pengguna Amazon EC2 untuk Instans Linux](#).

Izin Amazon EC2 untuk grup pengguna

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

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

- Administrator sistem — Membutuhkan izin untuk membuat dan mengelola instans AMIs, snapshot, volume, grup keamanan, dan sebagainya. Nikki melampirkan kebijakan yangAmazonEC2FullAccessAWS dikelola 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 memanggilDescribeInstances,RunInstances,StopInstances,StartInstances, danTerminateInstances.

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.

- Support pengguna — Tidak dapat melaksanakan tindakan Amazon EC2 kecuali mencantumkan sumber daya Amazon EC2 yang saat ini tersedia. Oleh karena itu, Nikki membuat dan melampirkan kebijakan ke grup pengguna Support yang hanya memungkinkan mereka menghubungi operasi API “Describe” Amazon EC2

Untuk contoh seperti apa kebijakan ini, lihat [Contoh kebijakan berbasis identitas IAM](#) dan [Menggunakan AWS Identity and Access Management](#) dalam Panduan Pengguna Amazon EC2 untuk Instans Linux.

Perubahan fungsi pekerjaan pengguna

Pada suatu masa, salah satu pengembang, Paulo Santos, mengubah fungsi pekerjaan dan menjadi manajer. Sebagai manajer, Paulo menjadi bagian dari grup pengguna Support sehingga ia dapat membuka kasus dukungan untuk pengembangnya. Mateo memindahkan Paulo dari grup pengguna Developer ke grup pengguna Support. Sebagai hasil dari langkah ini, kemampuannya untuk berinteraksi dengan instans Amazon EC2 sudah 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 AWS-s3-bucket.

Pembuatan pengguna dan grup pengguna lain

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

```
/aws-s3-bucket
  /home
    /zhang
    /major
  /share
    /developers
    /managers
```


Mateo membagi `/aws-s3-bucket` ke dalam satu set direktori rumah untuk setiap karyawan, dan area bersama untuk grup developer dan manajer.

Kini Mateo membuat serangkaian kebijakan untuk menetapkan izin kepada pengguna dan grup pengguna:

- Akses direktori beranda untuk Zhang — Mateo melampirkan kebijakan ke Wei yang memungkinkannya dia untuk membaca, menulis, dan mencantumkan objek apa pun yang dengan key prefix Amazon S3 `/aws-s3-bucket/home/zhang/`
- Akses direktori beranda untuk Major — Mateo melampirkan kebijakan ke Mary yang memungkinkannya dia untuk membaca, menulis, dan mencantumkan objek apa pun yang dengan key prefix Amazon S3 `/aws-s3-bucket/home/major/`
- Akses direktori bersama untuk grup pengguna developer — Mateo melampirkan kebijakan kepada grup pengguna yang memungkinkan developer untuk membaca, menulis dan mencantumkan objek dalam `/aws-s3-bucket/share/developers/`
- Akses direktori bersama untuk grup pengguna manajer — Mateo melampirkan kebijakan kepada grup pengguna yang memungkinkan manajer untuk membaca, menulis dan mencantumkan objek dalam `/aws-s3-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 seperti apa kebijakan ini, lihat [Kontrol Akses](#) dalam Panduan Pengguna Amazon Simple Storage Service. Untuk informasi tentang bagaimana kebijakan dievaluasi pada waktu aktif, lihat [Logika evaluasi kebijakan](#).

Perubahan fungsi pekerjaan pengguna

Pada suatu masa, 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 grupManagers pengguna, tetapi tidak lagi dapat mengakses data dalamshare/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 seorang karyawan Widget Company bernama Shirley Rodriguez perlu memasukkan data ke dalam bucket untuk digunakan oleh Example Corp. Nikki membuat grup pengguna yang disebut WidgetCodan nama penggunaShirley dan menambahkan Shirley ke grup WidgetCo pengguna. Nikki juga menciptakan bucket khusus yang disebut aws-s3-bucket untuk digunakan oleh Shirley.

Nikki memperbarui kebijakan yang ada atau menambahkan kebijakan baru untuk mengakomodasi mitra Widget Company. Misalnya, Nikki dapat membuat kebijakan baru yang menyangkal grup WidgetCo pengguna kemampuan 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 ini 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: Berikan akses ke konsol penagihan](#)
- [Tutorial IAM: Delegasikan akses AWS yang menggunakan peran IAM](#)
- [Tutorial IAM: Buat dan lampirkan kebijakan yang dikelola pelanggan pertama Anda](#)
- [Tutorial IAM: Menentukan izin untuk mengakses sumber daya AWS berdasarkan tanda](#)
- [Tutorial IAM: Izinkan pengguna untuk mengelola kredensi dan pengaturan MFA mereka](#)

Tutorial IAM: Berikan akses ke konsol penagihan

Akun AWS Pemilik ([Pengguna root akun AWS](#)) dapat memberikan pengguna IAM dan peran akses ke AWS Billing and Cost Management data untuk mereka Akun AWS. Instruksi dalam tutorial ini membantu Anda mengatur skenario yang telah diuji sebelumnya. Skenario ini membantu Anda memperoleh pengalaman langsung dalam mengonfigurasi izin penagihan tanpa perlu memengaruhi akun produksi AWS utama Anda.

[Prasyarat](#)

Buat persiapan berikut sebelum melakukan langkah-langkah dalam tutorial ini:

- Buat tes Akun AWS.
- Masuk ke pengujian Anda Akun AWS sebagai pengguna root.
- Catat Akun AWS jumlah akun pengujian Anda sehingga Anda dapat menggunakannya dalam tutorial. Dalam tutorial ini kita menggunakan contoh nomor akun 111122223333. Setiap kali langkah menggunakan nomor akun itu, gantilah dengan nomor akun pengujian Anda.

Langkah 1: Aktifkan akses IAM ke informasi penagihan di akun pengujian Anda AWS

Dalam skenario ini, Anda masuk ke pengujian Akun AWS sebagai pengguna root untuk memberikan akses IAM ke informasi penagihan. Saat Anda memberikan akses IAM ke informasi penagihan, pengguna dan peran IAM dapat mengakses konsol. AWS Billing and Cost Management Pengaturan ini tidak memberikan pengguna IAM dan peran izin yang diperlukan untuk halaman konsol ini, ini memungkinkan akses untuk pengguna IAM atau peran yang memiliki kebijakan IAM yang diperlukan. Jika kebijakan sudah dilampirkan ke pengguna atau peran IAM, tetapi setelah ini tidak diaktifkan, izin yang diberikan oleh kebijakan tersebut tidak berlaku.

Note

Akun AWS dibuat menggunakan AWS Organizations memiliki akses IAM ke informasi penagihan diaktifkan secara default.

Langkah 2: Buat pengguna dan grup uji

Dalam skenario ini, Anda memberi pengguna IAM akses ke konsol penagihan dan Anda membuat dua pengguna:

- Tepuk Candella

Pat adalah anggota departemen keuangan dan bekerja dengan penagihan dan pembayaran. Pat membutuhkan akses penuh ke informasi penagihan di Akun AWS.

- Terry Whitlock

Terry adalah bagian dari departemen dukungan TI Anda. Sebagian besar waktu Terry tidak memerlukan akses ke konsol penagihan, tetapi terkadang membutuhkan akses untuk menjawab pertanyaan bagi karyawan di departemen keuangan.

Langkah 3: Buat peran untuk memberikan akses ke AWS Billing konsol

Sebuah role IAM adalah identitas IAM yang dapat Anda buat di akun Anda yang memiliki izin spesifik. 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 dapat menjadi dapat diasumsikan oleh siapa pun yang membutuhkannya. Selain itu, peran tidak memiliki kredensi jangka panjang standar seperti kata sandi atau kunci akses yang terkait dengannya. Sebagai gantinya, saat Anda mengambil peran, kredensial keamanan sementara untuk sesi

peran Anda akan diberikan. Anda dapat menggunakan peran untuk mendelegasikan akses ke pengguna, aplikasi, atau layanan yang biasanya tidak memiliki akses ke sumber daya AWS Anda. Dalam skenario ini Anda membuat peran yang dapat diasumsikan oleh Terry Whitlock untuk mengakses konsol penagihan.

Langkah 4: Uji akses ke konsol

Setelah Anda menyelesaikan tugas-tugas inti, Anda siap untuk menguji kebijakan. Pengujian memastikan bahwa kebijakan bekerja dengan cara yang Anda inginkan. Dengan menguji akses setiap pengguna, Anda dapat membandingkan pengalaman pengguna.

Prasyarat

Buat persiapan berikut sebelum melakukan langkah-langkah dalam tutorial ini:

- Buat tesAkun AWS.
- Masuk ke pengujian Anda Akun AWS sebagai pengguna root.
- Catat Akun AWS jumlah akun pengujian Anda sehingga Anda dapat menggunakannya dalam tutorial. Dalam tutorial ini kita menggunakan contoh nomor akun 111122223333. Setiap kali langkah menggunakan nomor akun itu, gantilah dengan nomor akun pengujian Anda.

Langkah 1: Aktifkan akses IAM ke informasi penagihan di akun pengujian Anda AWS

Dalam skenario ini, Anda masuk ke pengujian Akun AWS sebagai pengguna root untuk memberikan akses IAM ke informasi penagihan. Saat Anda memberikan akses ke informasi penagihan, pengguna dan peran IAM dapat mengakses konsol. AWS Billing and Cost Management Pengaturan ini tidak memberikan pengguna IAM dan peran izin yang diperlukan untuk halaman konsol ini, hanya memungkinkan akses untuk pengguna IAM atau peran yang memiliki kebijakan IAM yang diperlukan.

Note

Akun AWSdibuat menggunakan AWS Organizations memiliki akses IAM ke informasi penagihan diaktifkan secara default.

Untuk mengaktifkan akses pengguna dan peran IAM ke konsol Manajemen Penagihan dan Biaya

1. Masuk ke AWS Management Console dengan kredensi 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 Update (Perbarui).

Halaman menampilkan pesan IAM pengguna/akses peran ke informasi penagihan diaktifkan.

Pada langkah berikutnya dari tutorial ini Anda melampirkan kebijakan IAM untuk memberikan atau menolak akses ke fitur penagihan tertentu.

Langkah 2: Buat pengguna dan grup uji

AWS Akun pengujian Anda tidak memiliki identitas apa pun yang ditentukan kecuali untuk pengguna root. Untuk memberikan akses ke informasi penagihan, kami membuat identitas tambahan kepada siapa kami dapat memberikan izin untuk mengakses informasi penagihan.


Buat pengguna dan grup uji

1. Masuk ke [Konsol IAM](#) sebagai pemilik akun dengan memilih Pengguna akar dan masukkan alamat email Akun AWS Anda. Di laman berikutnya, masukkan kata sandi Anda.

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 AWS Management Console sebagai pengguna root](#) di Panduan AWS Sign-In Pengguna.

2. Di panel navigasi, pilih Pengguna dan kemudian pilih Tambah pengguna.

 Note

Jika Anda mengaktifkan Pusat Identitas IAM, akan AWS Management Console menampilkan pengingat bahwa yang terbaik adalah mengelola akses pengguna di Pusat Identitas IAM. Dalam tutorial ini, pengguna IAM yang kita buat adalah belajar tentang menyediakan akses ke informasi penagihan. Jika Anda telah membuat pengguna di Pusat Identitas IAM, Anda menetapkan izin Penagihan yang disetel ke pengguna atau grup tersebut menggunakan Pusat Identitas IAM, bukan IAM.

3. Untuk Nama pengguna, masukkan **pcandella**. Nama tidak dapat berisi spasi.
4. Pilih kotak pilih di sebelah Berikan akses pengguna ke AWS Management Console — opsional dan kemudian pilih ingin membuat pengguna IAM.
5. Di bawah Kata sandi konsol, pilih Kata sandi yang dibuat otomatis.
6. Kosongkan kotak pilih di sebelah Pengguna harus membuat kata sandi baru saat masuk berikutnya (disarankan) dan kemudian pilih Berikutnya. Karena pengguna IAM ini untuk pengujian, kami akan mengunduh kata sandi untuk digunakan selama prosedur verifikasi.
7. Pada halaman Setel izin, di bawah opsi Izin, pilih Tambahkan pengguna ke grup. Kemudian, di bawah Grup pengguna, pilih Buat grup.
8. Pada halaman Buat grup pengguna, di Nama grup pengguna, masukkan **BillingGroup**. Kemudian, di bawah Kebijakan izin, pilih kebijakan fungsi pekerjaan AWS terkelola Penagihan.
9. Pilih Buat grup pengguna untuk kembali ke halaman Setel izin.
10. Di bawah Grup pengguna, pilih kotak pilih yang **BillingGroup** Anda buat.
11. Pilih Berikutnya untuk melanjutkan ke halaman Tinjauan dan buat.
12. Pada halaman Tinjau dan buat, tinjau daftar keanggotaan grup pengguna untuk pengguna baru. Ketika Anda siap untuk melanjutkan, pilih Buat pengguna.
13. Pada halaman Ambil kata sandi, pilih Unduh file.csv untuk menyimpan file.csv dengan informasi login pengguna (URL koneksi, nama pengguna, dan kata sandi).

Simpan file ini untuk digunakan sebagai referensi saat Anda masuk AWS sebagai pengguna IAM ini
14. Pilih Kembali ke daftar pengguna
15. Ulangi prosedur ini menggunakan modifikasi berikut untuk membuat pengguna untuk Terry Whitlock dan grup untuk pengguna dukungan.

- a. Pada langkah 3, untuk Nama pengguna, masukkan **twhitlock**.
- b. Pada langkah 8, untuk nama grup Pengguna, masukkan **SupportGroup**. Kemudian, di bawah Kebijakan izin, pilih kebijakan fungsi AWS pekerjaan terkelola. SupportUser

Anda dapat meninjau pengguna, grup, dan peran IAM baru di daftar konsol. Untuk setiap item yang Anda buat, Anda dapat memilih nama untuk melihat detailnya. Saat Anda melihat detail pengguna, konsol akan menampilkan Penagihan yang tercantum di bawah Kebijakan Izin untuk **pcandella** dan SupportUser tercantum di bawah Kebijakan Izin untuk **twhitlock**

Untuk informasi selengkapnya tentang penggunaan kebijakan untuk memberikan akses kepada pengguna IAM ke AWS Billing and Cost Management fitur, lihat [Menggunakan kebijakan berbasis identitas \(kebijakan IAM\) untuk AWS Billing di Panduan Pengguna](#). AWS Billing

Langkah 3: Buat peran untuk memberikan akses ke AWS Billing konsol

Anda dapat menggunakan peran untuk memberi pengguna IAM akses ke konsol penagihan. Peran menyediakan kredensi sementara yang dapat diasumsikan pengguna saat diperlukan. Dalam tutorial ini, pengguna **twhitlock** harus dapat mengakses informasi penagihan ketika permintaan dukungan dari departemen keuangan mengharuskan dia menyelidiki masalah.

1. Masuk ke [Konsol IAM](#) sebagai pemilik akun dengan memilih Pengguna akar dan masukkan alamat email Akun AWS Anda. Di laman berikutnya, masukkan kata sandi Anda.

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 AWS Management Console sebagai pengguna root](#) di Panduan AWS Sign-In Pengguna.

2. Di panel navigasi, pilih Pengguna dan kemudian pilih **twhitlock** pengguna untuk melihat detail pengguna. Salin ARN untuk **twhitlock** pengguna ke clipboard.
3. Di panel navigasi, pilih Peran lalu pilih Buat peran.
4. Pada halaman Pilih entitas tepercaya, pilih Kebijakan kepercayaan kustom dan kemudian di bawah Edit pernyataan lengkapi item berikut:

- Tambahkan tindakan untuk STS - Verifikasi AssumeRole yang dipilih.
 - Tambahkan pilihan utama Tambahkan untuk menampilkan kotak dialog Add principal. Untuk tipe Principal pilih pengguna IAM lalu untuk ARN tempel ARN untuk pengguna twhitlock yang Anda salin ke clipboard pada langkah 16. Kemudian pilih Tambahkan prinsipal.
5. Pilih Berikutnya untuk pergi ke halaman Tambahkan izin.
 6. Di bawah Kebijakan izin di kotak filter, masukkan **Billing** lalu pilih kebijakan fungsi AWS pekerjaan terkelola Penagihan.
 7. Pilih Berikutnya untuk pergi ke halaman Nama, ulasan, dan buat. Di bawah Nama peran, masukkan **TempBillingAccess** lalu pilih Buat peran.

Anda diberi tahu bahwa peran telah dibuat. Lihat peran untuk menampilkan detail tentang peran tersebut. Di bagian Ringkasan perhatikan informasi berikut:

- Durasi sesi maksimum adalah 1 jam secara default. Setelah waktu itu pengguna yang mengambil peran kembali ke izin akun dasar mereka. Jika pengguna ingin terus menggunakan izin peran, mereka harus beralih peran lagi. Anda dapat mengedit peran untuk meningkatkan durasi maksimum. Durasi sesi terpanjang yang mungkin adalah 12 jam.
- Tautan untuk beralih peran di konsol. Anda dapat menyalin tautan untuk memberikannya langsung kepada pengguna yang Anda tambahkan sebagai prinsipal dalam kebijakan kepercayaan. Anda dapat melihat dan mengedit kebijakan kepercayaan dari tab Hubungan kepercayaan.

Langkah 4: Uji akses ke konsol

Kami menyarankan Anda menguji akses dengan masuk sebagai pengguna pengujian untuk mempelajari apa yang mungkin dialami pengguna Anda. Gunakan langkah-langkah berikut untuk masuk menggunakan kedua akun uji untuk melihat perbedaan antara hak akses.

Untuk menguji akses penagihan dengan masuk dengan kedua pengguna uji

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 masuk AWS menggunakan cookie browser untuk mengingat nama pengguna dan informasi akun pengguna IAM. 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 menyetikkan ID AWS akun atau alias akun Anda untuk diarahkan ke halaman login pengguna IAM untuk akun Anda.

2. Masuk dengan setiap pengguna menggunakan langkah-langkah yang disediakan di bawah ini sehingga Anda dapat membandingkan pengalaman pengguna yang berbeda.

Akses penuh

- a. Masuk ke Anda Akun AWS sebagai pengguna **pcandella**.
- b. Pada bilah navigasi, pilih pcandella @111122223333, lalu pilih Dasbor Penagihan.
- c. Jelajahi halaman dan pilih berbagai tombol untuk memastikan bahwa Anda memiliki izin modifikasi penuh.

Tidak ada akses

- a. Masuk ke Anda Akun AWS sebagai pengguna **twhitlock**.
- b. Pada bilah navigasi, pilih twhitlock @111122223333, lalu pilih Dasbor Penagihan.
- c. Sebuah pesan menampilkan menyatakan Anda memerlukan izin. Tidak ada data penagihan yang terlihat.

Beralih peran untuk meningkatkan akses

- a. Masuk ke Anda Akun AWS sebagai pengguna **twhitlock**.
- b. Pada bilah navigasi, pilih twhitlock @111122223333, lalu pilih Beralih peran.

Halaman peran Switch terbuka. Lengkapi informasinya sebagai berikut:

- Akun -111122223333
- Peran - **TempBillingAccess**

Pilih Beralih peran

Atau, Anda dapat menggunakan URL yang disediakan di Link untuk beralih peran di konsol untuk membuka halaman peran Switch.

- c. Konsol menampilkan AWS BillingDasbor dan bilah navigasi menampilkan TempBillingAccess@111122223333.

Ringkasan

Anda sekarang telah menyelesaikan langkah-langkah yang diperlukan untuk memberikan pengguna IAM akses ke AWS Billing konsol. Akibatnya, Anda telah melihat secara langsung seperti apa pengalaman konsol penagihan pengguna Anda. Sekarang Anda dapat melanjutkan untuk menerapkan logika ini di lingkungan produksi Anda kapan saja.

Sumber daya terkait

Untuk informasi terkait yang ditemukan di Panduan AWS Billing Pengguna, lihat sumber daya berikut:

- [Mengaktifkan Akses ke konsol AWS Billing](#)
- [AWSContoh kebijakan penagihan](#)
- [Menggunakan kebijakan berbasis identitas \(kebijakan IAM\) untuk Penagihan AWS](#)
- [Memigrasi kontrol akses untuk AWS Billing](#)

Untuk informasi terkait di Panduan Pengguna IAM, lihat sumber daya berikut:

- [Kebijakan terkelola dan kebijakan inline](#)
- [Mengontrol akses pengguna IAM keAWS Management Console](#)
- [Melampirkan kebijakan ke grup pengguna IAM](#)

Tutorial IAM: Delegasikan aksesAWSakun yang menggunakan peran IAM

Tutorial ini mengajarkan Anda bagaimana menggunakan peran untuk mendelegasikan akses ke sumber daya yang berbedaAkun AWSbawa Anda sendiri disebutProduksidanPengembangan. 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. Setelah mengonfigurasi peran, Anda melihat cara menggunakan peran dari AWS Management Console, AWS CLI, dan API.

Note

Peran IAM dan kebijakan berbasis sumber daya mendelegasikan akses ke seluruh akun hanya dalam satu partisi tunggal. Misalnya, anggap Anda memiliki akun di AS Barat (N. California) dalam partisi `aws` standar. Anda juga memiliki akun di Tiongkok (Beijing) dalam partisi `aws-cn`. Anda tidak dapat menggunakan kebijakan berbasis sumber daya Amazon S3 di akun Anda di Tiongkok (Beijing) untuk memungkinkan akses bagi pengguna dalam akun `aws` standar Anda.

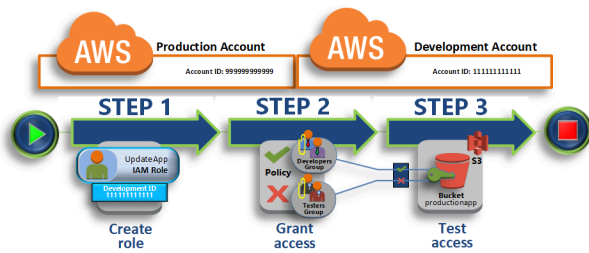
Dalam tutorial ini, `Produksi` akun mengelola aplikasi langsung. Pengembang dan penguji menggunakan `Pengembangan` akun sebagai kotak pasir untuk menguji aplikasi secara bebas. Di setiap akun, Anda menyimpan informasi aplikasi di bucket Amazon S3. Anda mengelola pengguna IAM di `Pengembangan` akun, di mana Anda memiliki dua grup pengguna IAM: `Pengembang` dan `Penguji`. Pengguna di kedua grup pengguna memiliki izin untuk bekerja di akun `Pengembangan` dan mengakses sumber daya di sana. Dari waktu ke waktu, pengembang harus memperbarui aplikasi langsung di `Produksi` akun. Pengembang menyimpan aplikasi ini dalam bucket Amazon S3 yang disebut `productionapp`.

Di akhir tutorial ini, Anda memiliki hal berikut:

- Pengguna di `Pengembangan` akun (akun tepercaya) diizinkan untuk mengambil peran tertentu dalam `Produksi` akun.
- Peran dalam `Produksi` akun (akun kepercayaan) diizinkan untuk mengakses bucket Amazon S3 tertentu.
- Yang `productionapp` pember di `Produksi` akun.

Pengembang dapat menggunakan peran dalam `AWS Management Console` untuk mengakses `productionapp` pember di `Produksi` akun. Mereka juga dapat mengakses bucket dengan menggunakan panggilan API yang diautentikasi oleh kredensi sementara yang disediakan oleh peran tersebut. Percobaan serupa oleh `Penguji` untuk menggunakan peran tersebut gagal.

Alur kerja ini memiliki tiga langkah dasar:



Buat peran dalam Akun Produksi

Pertama, Anda menggunakan AWS Management Console untuk membangun kepercayaan antara Akun Produksi (nomor ID 999999999999) dan Akun Pengembangan (nomor ID 111111111111). Anda mulai dengan membuat peran IAM bernama `UpdateApp`. Saat Anda membuat peran, Anda menentukan Akun Pengembangan sebagai entitas tepercaya dan tentukan kebijakan izin yang memungkinkan pengguna tepercaya untuk memperbarui `productionapp`.

Berikan akses ke peran tersebut

Di bagian ini, Anda mengubah kebijakan grup pengguna IAM untuk menolak akses Penguji ke `UpdateApp` peran. Karena Penguji memiliki `PowerUser` akses dalam skenario ini, dan Anda harus secara eksplisit menyangkalkan kemampuan untuk menggunakan peran.

Akses uji dengan mengalihkan peran

Akhirnya, sebagai Pengembang, Anda menggunakan `UpdateApp` peran untuk memperbarui `productionapp` di Akun Produksi. Anda melihat cara mengakses peran melalui konsol AWS, AWS CLI, dan API.

Prasyarat

Tutorial ini mengasumsikan bahwa Anda telah melakukan hal berikut:

- Dua memisahkan Akun AWS yang dapat Anda gunakan, satu untuk mewakili Akun Pengembangan, dan satu untuk mewakili Akun Produksi.
- Pengguna dan grup pengguna di Akun Pengembangan dibuat dan dikonfigurasi sebagai berikut:

Pengguna	Grup pengguna	Izin
David	Developer	Kedua pengguna dapat masuk dan menggunakan AWS Management Console di dalam Akun Pengembangan.

Pengguna	Grup pengguna	Izin
Jane	Penguji	

- Anda tidak memerlukan pengguna atau grup pengguna yang dibuat diProduksiakun.
- Bucket Amazon S3 yang dibuat diProduksiakun. Anda bisa menyebutnyaProductionAppdalam tutorial ini, tetapi karena nama bucket S3 harus unik secara global, Anda harus menggunakan bucket dengan nama yang berbeda.

Buat peran dalam Akun Produksi

Anda dapat mengizinkan pengguna dari satuAkun AWSuntuk mengakses sumber daya di lainAkun AWS. Untuk melakukannya, buat peran yang menentukan siapa yang dapat mengaksesnya dan izin apa yang diberikan kepada pengguna yang dialihkan.

Dalam langkah tutorial ini, Anda membuat peran dalamProduksiakun dan tentukanPengembanganakun sebagai entitas tepercaya. Anda juga membatasi izin peran untuk hanya membaca dan menulis akses keproductionappmember. Siapa pun yang diberi izin untuk menggunakan peran dapat membaca dan menulis keproductionappmember.

Sebelum Anda dapat membuat peran, Anda memerlukanID akundariPengembangan Akun AWS. Masing-masingAkun AWSmemiliki pengenalan ID akun unik yang ditetapkan padanya.

Untuk mendapatkan PengembanganAkun AWSID

1. Masuk keAWS Management Consolesebagai administratorPengembanganakun, dan buka konsol IAM di<https://console.aws.amazon.com/iam/>.
2. Di bilah navigasi, pilih Dukungan, kemudian Pusat Dukungan. Nomor akun (ID) 12 digit Anda yang masuk saat ini muncul di panel navigasi Pusat Dukungan. Untuk skenario ini, Anda dapat menggunakan ID akun 111111111111 untukPengembanganakun. Namun, Anda harus menggunakan ID akun yang valid jika menggunakan skenario ini di lingkungan pengujian.

Untuk membuat peran dalam akun produksi yang dapat digunakan olehPengembanganakun

1. Masuk keAWS Management Consolesebagai administratorProduksiakun, dan buka konsol IAM.
2. Sebelum membuat peran, siapkan kebijakan terkelola yang menentukan 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 `productionapp`.

Meskipun AWS menyediakan beberapa kebijakan terkelola Amazon S3, tidak ada yang menyediakan akses baca dan tulis ke bucket Amazon S3 tunggal. Anda dapat membuat kebijakan Anda sendiri.

Pada panel navigasi, pilih Kebijakan lalu pilih Buat kebijakan.


3. Pilih tab JSON dan salin teks dari dokumen kebijakan JSON berikut. Tempelkan teks ini ke JSON kotak teks, mengganti ARN sumber daya (`arn:aws:s3:::productionapp`) dengan yang asli untuk bucket Amazon S3 Anda.

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

Yang `ListAllMyBuckets` tindakan memberikan izin untuk mencantumkan semua bucket yang dimiliki oleh pengirim permintaan yang diautentikasi. Izin `ListBucket` memungkinkan

pengguna melihat objek dalam bucket `productionapp`. Izin `GetObject`, `PutObject`, `DeleteObject` memungkinkan pengguna melihat, memperbarui, dan menghapus konten di bucket `productionapp`.

4. Mengatasi peringatan keamanan, kesalahan, atau peringatan umum yang dihasilkan selama [validasi kebijakan](#), dan kemudian pilih `Berikutnya`.

 Note

Anda dapat beralih di antara `Visual` dan `JSON` opsi editor kapan saja. Namun, jika Anda membuat perubahan atau memilih `Berikutnya` di dalam `Visual` editor, IAM mungkin merestrukturisasi kebijakan Anda untuk mengoptimalkannya untuk editor visual. Untuk informasi selengkapnya, lihat [Restrukturisasi kebijakan](#).

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

Kebijakan baru muncul di daftar kebijakan terkelola.

6. Di panel navigasi, pilih `Peran` lalu pilih `Buat` peran.
7. Pilih `Sebuah Akun AWS` jenis peran.
8. Untuk `ID` Akun, ketik `PengembanganID` akun.

Tutorial ini menggunakan contoh `ID` akun `111111111111` untuk `Pengembangan` akun. 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 [Menggunakan otentikasi multi-faktor \(MFA\) di AWS](#).

9. Pilih `Berikutnya`: Izin untuk mengatur izin yang terkait dengan peran.
10. Centang kotak di samping kebijakan yang Anda buat sebelumnya.

i Kiat

Untuk Menyaring, pilih Pelanggan dikelola untuk memfilter daftar agar hanya menyertakan kebijakan yang Anda buat. Ini menyembunyikan AWS menciptakan kebijakan dan membuatnya lebih mudah untuk menemukan yang Anda butuhkan.

Lalu, pilih Selanjutnya.

11. (Opsional) Tambahkan metadata ke peran dengan melampirkan tag sebagai pasangan kunci-nilai. Untuk informasi selengkapnya tentang menggunakan tanda di IAM, lihat [Menandai sumber daya IAM](#).
12. (Opsional) Untuk Deskripsi, masukkan deskripsi untuk peran baru.
13. Setelah meninjau peran, klik Buat peran.

Peran UpdateApp muncul di daftar peran.

Sekarang Anda harus mendapatkan Amazon Resource Name (ARN) peran, pengenal unik untuk peran tersebut. Saat mengubah kebijakan grup pengguna Pengembang dan Penguji, Anda menentukan ARN peran untuk memberikan atau menolak izin.

Untuk mendapatkan ARN untuk UpdateApp

1. Di panel navigasi konsol IAM, pilih Peran.
2. Dalam daftar peran, pilih peran UpdateApp.
3. Di bagian Ringkasan di panel perincian, salin nilai ARN Peran.

Akun Produksi memiliki ID akun 999999999999, jadi peran ARN adalah `arn:aws:iam::999999999999:role/UpdateApp`. Pastikan bahwa Anda memberikan yang nyata Akun AWSID untuk akun Produksi.

Pada titik ini, Anda telah membangun kepercayaan antara Produksi dan Pengembangan akun. Anda melakukan ini dengan menciptakan peran dalam Produksi akun yang mengidentifikasi Pengembangan akun sebagai kepala sekolah tepercaya. Anda juga mendefinisikan apa yang pengguna yang beralih ke UpdateApp peran dapat dilakukan.

Selanjutnya, ubah izin untuk grup pengguna.

Berikan akses ke peran tersebut

Pada titik ini, anggota grup pengguna Penguji dan Pengembang memiliki izin yang memungkinkan mereka untuk menguji aplikasi secara bebas di Pengembangan akun. Gunakan langkah-langkah berikut yang diperlukan untuk menambahkan izin untuk memungkinkan beralih ke peran.

Untuk memodifikasi grup pengguna Pengembang untuk memungkinkan mereka beralih ke UpdateApp peran

1. Masuk sebagai administrator di Pengembangan akun, dan buka konsol IAM.
2. Pilih User groups (Grup pengguna), lalu pilih Developers.
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 mengizinkan tindakan AssumeRole pada peran UpdateApp di akun Produksi. Pastikan bahwa Anda berubah **PRODUKSI-AKUN-ID** di dalam Resource elemen untuk yang sebenarnya Akun AWSID akun Produksi.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::PRODUCTION-ACCOUNT-ID:role/UpdateApp"
  }
}
```

Efek Allow secara eksplisit memungkinkan grup Developer mengakses peran UpdateApp dalam akun Produksi. Setiap pengembang yang mencoba mengakses peran berhasil.

6. Pilih Tinjau kebijakan.
7. Ketik aName seperti **allow-assume-S3-role-in-production**.
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 izin "Deny" ke grup Penguji untuk memastikan mereka tidak dapat menjalankan peran tersebut. Jika Anda tidak memerlukan prosedur ini di lingkungan Anda, maka sebaiknya Anda tidak menambahkannya. "Deny" izin membuat gambaran izin keseluruhan lebih rumit untuk dikelola dan dipahami. Gunakan "Deny" izin hanya jika Anda tidak memiliki pilihan yang lebih baik.

Untuk memodifikasi grup pengguna penguji untuk menolak izin untuk mengasumsikan **UpdateApp** peran

1. Pilih User groups (Grup pengguna), lalu pilih Testers (Penguji).
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 UpdateApp. Pastikan bahwa Anda berubah **PRODUKSI-AKUN-ID** di dalam Resource elemen untuk yang sebenarnya Akun AWSID akun Produksi.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Deny",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::PRODUKSI-ACCOUNT-ID:role/UpdateApp"
  }
}
```

Efek Deny secara eksplisit menolak akses grup Penguji ke peran UpdateApp di akun Produksi. Setiap penguji yang mencoba mengakses peran menerima pesan akses ditolak.

5. Pilih Tinjau kebijakan.
6. Ketik **anamasukadeny-assume-s3-role-in-production**.
7. Pilih Buat kebijakan.

Grup pengguna Developer sekarang memiliki izin untuk menggunakan peran UpdateApp di akun Produksi. Grup pengguna Penguji dicegah dari menggunakan peran UpdateApp.

Selanjutnya, Anda dapat melihat bagaimana David, seorang pengembang, dapat mengakses `productionapp` pember di akun Produksi. David dapat mengakses bucket dari AWS Management Console, AWS CLI, atau API AWS.

Akses uji dengan mengalihkan peran

Setelah menyelesaikan dua langkah pertama dari tutorial ini, Anda memiliki peran yang memberikan akses ke sumber daya di `Produksi` akun. Anda juga memiliki satu grup pengguna di `Pengembangan` akun dengan pengguna diizinkan untuk menggunakan peran itu. Langkah ini membahas cara menguji pengalihan ke peran tersebut dari AWS Management Console, AWS CLI, dan API AWS.

Important

Anda dapat beralih ke peran hanya setelah masuk sebagai pengguna IAM atau pengguna federasi. Selain itu, jika Anda meluncurkan instans Amazon EC2 untuk menjalankan aplikasi, aplikasi dapat mengambil peran melalui profil instansnya. Anda tidak dapat beralih ke peran saat Anda masuk sebagai Pengguna root akun AWS.

Beralih peran (konsol)

Jika David perlu bekerja di dalam `Produksi` lingkungan di AWS Management Console, dia bisa melakukannya dengan menggunakan `Beralih 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 `productionappbucket`, tetapi tidak dapat bekerja dengan sumber daya lain di `Produksi`. Sementara David menggunakan peran itu, ia juga tidak dapat menggunakan hak pengguna kekuatannya di `Pengembangan` akun. Itu karena hanya satu set izin yang dapat berlaku pada satu waktu.

Important

Beralih peran menggunakan AWS Management Console hanya bekerja 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 hanya dapat mengakses akun Anda menggunakan API AWS atau alat baris perintah. Pihak ketiga tidak dapat menggunakan konsol karena tidak dapat menyediakan nilai untuk `ExternalId`. Untuk informasi selengkapnya tentang skenario

ini, lihat [Cara menggunakan ID eksternal saat memberikan akses ke sumber daya AWS Anda kepada pihak ketiga](#), dan [Cara Mengaktifkan Akses Lintas Akun keAWS Management Console](#) di Blog AWS Security.

IAM menyediakan dua cara yang dapat digunakan David untuk memasuki Beralih Peran halaman:

- David menerima tautan dari administrator mereka yang menunjuk ke konfigurasi Switch Role 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. Semua David perlu lakukan adalah memilih Beralih Peran.
- 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 menandatangani keAWS Management Console menggunakan pengguna normalnya di Pengembang kelompok pengguna.
2. Mereka memilih tautan yang dikirim oleh administrator kepada mereka. Ini membawa David ke Beralih Peran halaman dengan ID akun atau alias dan informasi nama peran sudah diisi.

— atau —

David memilih nama mereka (menu Identity) pada bilah navigasi, dan kemudian memilih Beralih Peran.

Jika ini adalah pertama kalinya David mencoba mengakses halaman Switch Role dengan cara ini, dia pertama kali mendarat di run pertama Beralih Peran halaman. Halaman ini memberikan informasi tambahan tentang bagaimana peran pengalihan dapat mengizinkan pengguna untuk mengelola sumber daya di seluruh Akun AWS. David harus memilih Beralih Peran pada halaman ini untuk menyelesaikan sisa prosedur ini.

3. Selanjutnya, untuk mengakses peran, David harus mengetikkan nomor ID akun Produksi (999999999999) dan nama peran (UpdateApp) secara manual.

Juga, David ingin memantau peran dan izin terkait yang saat ini aktif di IAM. Untuk melacak informasi ini, dia mengetik PRODUCTION 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 UpdateAppnya memiliki izin.
5. Setelah selesai, David dapat kembali ke izin aslinya. Untuk melakukan itu, mereka memilih PRODUKSI nama tampilan peran pada bilah navigasi dan kemudian pilih Kembali ke David @ 111111111111.
6. Lain kali David ingin beralih peran dan memilih Identitas menu di bar navigasi, ia melihat entri PRODUKSI masih ada dari terakhir kali. Dia cukup memilih entri itu untuk segera beralih peran tanpa memasukkan kembali ID akun dan nama peran.

Beralih peran (AWS CLI)

Jika David perlu bekerja di lingkungan produksi di baris perintah, dia bisa 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 di variabel lingkungan sehingga perintah AWS CLI berikutnya bekerja menggunakan izin peran. Sementara David menggunakan peran tersebut, ia tidak dapat menggunakan hak pengguna kekuatannya di Pengembangan akun, 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 prompt perintah, dan mengonfirmasi klien AWS CLI 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 peran sakelar dengan menjalankan perintah berikut untuk beralih keUpdateAppperan dalamProduksiakun. Dia menerima peran ARN dari administrator yang menciptakan peran. Perintah ini mengharuskan Anda memberikan nama sesi, Anda juga dapat memilih teks apa pun yang Anda sukai untuk itu.

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

David kemudian melihat hal-hal berikut di output:

```
{
  "Credentials": {
    "SecretAccessKey": "wJa1rXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
    "SessionToken": "AQoDYXdzEGcaEXAMPLE2gsYULo
+Im5ZEXAMPLEEeYjs1M2FUIgIJx9tQqNMBEXAMPLE
CvSryh0FW7jEXAMPLEW+vE/7s1HRpXviG7b+qYf4nD00EXAMPLEmj4wxS04L/
uZEXAMPLECihzFB51TYLto9dyBgSDy
EXAMPLE9/
g7QRUhZp4bqbEXAMPLENwGPy0j59pFA41NKCIkVgkREXAMPLEj1zxQ7y52gekeVEXAMPLEDiB9ST3Uuysg
sKdEXAMPLE1TVastU1A0SKFEXAMPLEiYWCC/Cs8EXAMPLEEpZg0s+6hz4AP4KEXAMPLERbASP
+4eZScEXAMPLEsnf87e
NhyDHq6ikBQ==",
    "Expiration": "2014-12-11T23:08:07Z",
    "AccessKeyId": "AKIAIOSFODNN7EXAMPLE"
  }
}
```

3. David melihat tiga bagian yang mereka butuhkan di bagian Kredensial dari output.
 - AccessKeyId
 - SecretAccessKey
 - SessionToken

David perlu mengonfigurasi lingkungan AWS CLI untuk menggunakan parameter ini pada 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 tidak 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.

Note

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 Alat untuk Windows Powershell, lihat [Beralih ke peran IAM \(Alat untuk Windows PowerShell\)](#)

```
set AWS_ACCESS_KEY_ID=AKIAIOSFODNN7EXAMPLE
set AWS_SECRET_ACCESS_KEY=wJa1rXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
set AWS_SESSION_TOKEN=AQoDYXdzEGcaEXAMPLE2gsYULo
+Im5ZEXAMPLEeYjs1M2FUIgIJx9tQqNMBEXAMPLECvS
Ryh0FW7jEXAMPLEw+vE/7s1HRpXviG7b+qYf4nD00EXAMPLEmj4wxS04L/
uZEXAMPLEcihzFB51TYLto9dyBgSDyEXA
MPLEKEY9/
g7QRUhZp4bqbEXAMPLENwGPy0j59pFA41NKCIkVgkREXAMPLEj1zxQ7y52gekeVEXAMPLEDiB9ST3UusKd
EXAMPLE1TVastU1A0SKFEXAMPLEiywCC/Cs8EXAMPLEpZg0s+6hz4AP4KEXAMPLERbASP
+4eZScEXAMPLENhykxiHen
DHq6ikBQ==
```

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

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


```
aws s3 ls s3://productionapp
```

Karena nama bucket Amazon S3 unik secara universal, tidak perlu menentukan ID akun yang memiliki bucket. Untuk mengakses sumber daya untuk lainnya AWS layanan, lihat [AWS CLI dokumentasi](#) untuk layanan itu untuk perintah dan sintaks yang diperlukan untuk referensi sumber dayanya.

Menggunakan AssumeRole (AWS API)

Ketika David perlu membuat update ke `Produksi` akun dari kode, dia membuat `AssumeRole` panggilan untuk mengasumsikan `UpdateApp` peran. Panggilan mengembalikan mandat sementara yang dapat ia gunakan untuk mengakses `productionapp` pember di `Produksi` akun. Dengan kredensial tersebut, David dapat melakukan panggilan API untuk memperbarui bucket `productionapp`. Namun, ia tidak dapat melakukan panggilan API untuk mengakses sumber daya lain di `Produksi` akun, meskipun ia memiliki izin pengguna daya di `Pengembangan` akun.

Untuk mengambil peran

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

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.

2. Dengan kredensial sementara, David membuat panggilan `s3:PutObject` untuk memperbarui bucket `productionapp`. Mereka akan meneruskan kredensi ke panggilan API sebagai `AuthParams` parameter. Karena kredensial peran sementara hanya memiliki akses hanya baca dan tulis ke bucket `productionapp`, tindakan lainnya di akun `Produksi` ditolak.

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

Sumber daya terkait

- Untuk informasi selengkapnya tentang pengguna IAM dan grup pengguna, lihat [Identitas IAM \(pengguna, pengguna, dan peran\)](#).

- Untuk informasi selengkapnya tentang bucket Amazon S3, lihat [Buat Bucket](#) di dalam Panduan Pengguna Layanan Penyimpanan Sederhana Amazon.
- 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?](#).

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 grup pengguna untuk mengontrol pengguna IAM mana yang dapat mengakses peran tersebut. Akibatnya, pengembang dari Pengembangan akun dapat membuat update ke `production` app member di Produksi akun dengan menggunakan kredensi sementara.

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

Dalam tutorial ini, Anda menggunakan AWS Management Console 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 AWS Management Console 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 masuki 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 AWS Management Console 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.

```
{
  "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 di sebelah UsersReadOnlyAccessToIAMConsole 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 uji `PolicyUser` 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](#)
- [Mengontrol akses pengguna IAM keAWS Management Console](#)

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: Menentukan izin untuk mengakses sumber daya AWS berdasarkan tanda

Kontrol akses berbasis atribut (ABAC) adalah strategi otorisasi yang mendefinisikan izin berdasarkan atribut. Di AWS, atribut ini disebut tanda. Anda dapat melampirkan tanda ke sumber daya IAM, termasuk entitas IAM (pengguna atau peran) dan ke sumber daya AWS. Anda dapat menentukan kebijakan yang menggunakan kunci syarat tanda untuk memberikan izin ke 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 kebijakan AWS tradisional, yang mengharuskan Anda untuk mencantumkan setiap sumber daya individu. Untuk informasi selengkapnya tentang ABAC dan manfaatnya dibandingkan kebijakan tradisional, lihat [Untuk apa ABAC? AWS](#).

Note

Anda harus memberikan 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 melihat atau mengedit hanya sumber daya AWS yang hanya diperlukan untuk tugas 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 di kebijakan dengan setiap tindakan dan sumber daya layanan masing-masing, lihat [Tindakan, Sumber Daya, dan Kunci Kondisi untuk Layanan AWS](#). Anda dapat mengonfigurasi penyedia identitas web atau yang berbasis SAML untuk meneruskan [tanda sesi](#) ke AWS. Ketika karyawan Anda menggabungkan ke AWS, atribut mereka diterapkan pada prinsipal yang mereka hasilkan di 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 proyek Unicorn 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 mengharuskan tanda alokasi biaya cost-center mengaktifkan laporan penagihan AWS khusus. Untuk informasi selengkapnya, lihat [Penggunaan Tanda Alokasi Biaya](#) dalam Panduan Pengguna AWS Billing and Cost Management.

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.

Prasyarat

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

- Sebuah Akun AWS yang dapat Anda masuki 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 AWS Management Console, pilih Support pada bilah navigasi di kanan atas, lalu pilih Support Center. Nomor akun (ID) muncul di panel navigasi di sebelah kiri.



Support Center ✕

Account number: 123412341234

- Pengalaman membuat dan mengedit pengguna, peran, dan kebijakan IAM di AWS Management Console. Namun, jika Anda memerlukan bantuan mengingat proses manajemen IAM, tutorial ini menyediakan tautan di mana Anda dapat melihat step-by-step instruksi.

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 placeholder yang dicetak miring dengan informasi akun Anda.

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

```

    "Resource": "arn:aws:iam::account-ID-without-hyphens: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 [Membuat grup pengguna IAM](#) dan [Menambahkan dan menghapus pengguna di grup pengguna IAM](#).

2. Buat pengguna IAM berikut, lampirkan kebijakan `access-assume-role` izin. Pastikan Anda memilih Menyediakan akses pengguna ke AWS Management Console, dan kemudian menambahkan tag berikut. Untuk informasi selengkapnya tentang membuat dan mendai pengguna baru, lihat [Membuat pengguna IAM \(konsol\)](#).

Pengguna ABAC

Nama pengguna	Kunci tag pengguna	Nilai tag pengguna
access-Arn timer-peg-e ng	access-project	peg
	access-team	eng
	cost-center	987654
access-Mary-peg-qas	access-project	peg
	access-team	qas
	cost-center	987654

Nama pengguna	Kunci tag pengguna	Nilai tag pengguna
access-Saanvi-uni-eng	access-project	uni
	access-team	eng
	cost-center	123456
access-Carlos-uni-qas	access-project	uni
	access-team	qas
	cost-center	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: Mengizinkan 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 instans berdasarkan tanda](#)
- [IAM: Ambil peran yang memiliki tanda tertentu](#)

Kebijakan ABAC: Akses Sumber Daya Secrets Manager Hanya Saat Prinsipal dan Tag 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`.

```

{
  "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 [Tindakan, 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 [Kunci konteks multivaluasi](#).
- 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 [Membuat peran untuk mendelegasikan izin kepada pengguna IAM](#). Jika Anda memilih untuk menggunakan federasi daripada pengguna dan peran IAM, lihat [Tutorial IAM: Menggunakan tanda sesi SAML untuk ABAC](#).

Peran ABAC

Fungsi pekerjaan	Nama peran	Tanda peran	Deskripsi peran
Proyek Teknik Pegasus	access-peg-engineering	access-project = peg access-team = eng cost-center = 987654	Memungkinkan insinyur membaca semua sumber daya teknik dan membuat serta mengelola sumber daya teknik Pegasus.
Jaminan Kualitas Proyek Pegasus	access-peg-quality-assurance	access-project = peg	Memungkinkan tim QA untuk membaca semua sumber daya QA dan membuat, serta

Fungsi pekerjaan	Nama peran	Tanda peran	Deskripsi peran
		access-team = qas cost-center = 987654	mengelola semua sumber daya QA Pegasus.
Proyek Teknik Unicorn	access-uni-engineering	access-project = uni access-team = eng cost-center = 123456	Memungkinkan insinyur membaca semua sumber daya teknik dan membuat, serta mengelola sumber daya teknik Unicorn.
Jaminan Kualitas Proyek Unicorn	access-uni-quality-assurance	access-project = uni access-team = qas cost-center = 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 `access-Arnav-peg-eng` Pengguna 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 beralih peran di AWS Management Console, lihat [Beralih ke peran \(konsol\)](#)

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

Secrets Manager menampilkan peringatan bahwa Anda tidak memiliki izin untuk AWS layanan tambahan yang bekerja dengan Secrets Manager. Misalnya, untuk membuat kredensial basis data Amazon RDS, Anda harus memiliki izin untuk menjelaskan instans RDS, kluster RDS, dan kluster 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 tanda terakhir diizinkan dan akan berhasil membuat rahasia.

Kombinasi tanda ABAC untuk peran **test-access-peg-eng**

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	owner = 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	Name = Jane	Diizinkan karena ketiga tanda yang diperlukan ada dan nilainya sesuai dengan nilai-nilai peran. Anda juga diperbolehkan untuk menyertakan tanda Name .

- 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.

Peran dan tanda ABAC

Nama pengguna	Nama peran	Nama rahasia	Tanda rahasia
access-Mary-peg-qas	access-peg-quality-assurance	test-access-peg-qas	access-project = peg access-team = qas cost-center = 987654
access-Saanvi-uni-eng	access-uni-engineering	test-access-uni-eng	access-project = uni access-team = eng cost-center = 123456
access-Carlos-uni-qas	access-uni-quality-assurance	test-access-uni-qas	access-project = uni access-team = qas cost-center = 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 beralih peran di AWS Management Console, lihat [Beralih ke peran \(konsol\)](#).

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.

Rahasia ABAC melihat perilaku setiap peran

Nama peran	Nama rahasia	Perilaku yang diharapkan
access-peg-engineering	test-access-peg-eng	Diizinkan
	test-access-peg-qas	Ditolak
	test-access-uni-eng	Diizinkan

Nama peran	Nama rahasia	Perilaku yang diharapkan
	test-access-uni-qas	Ditolak
access-peg-quality-assurance	test-access-peg-eng	Ditolak
	test-access-peg-qas	Diizinkan
	test-access-uni-eng	Ditolak
access-uni-engineering	test-access-uni-qas	Diizinkan
	test-access-peg-eng	Diizinkan
	test-access-peg-qas	Ditolak
	test-access-uni-eng	Diizinkan
access-uni-quality-assurance	test-access-uni-qas	Ditolak
	test-access-peg-eng	Ditolak
	test-access-peg-qas	Diizinkan
	test-access-uni-eng	Ditolak
	test-access-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. Karena perusahaan Anda menambahkan proyek, tim, atau orang baru ke AWS, Anda tidak perlu memperbarui kebijakan berbasis ABAC 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 Peran 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 pengguna `access-Saanvi-uni-eng`.
7. Pada tab Izin, hapus kebijakan `access-assume-role` izin.
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 \(konsol\)](#).

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 multivalai. Anda tidak dapat menandai pengguna Saanvi dengan `access-project = peg` dan `access-`

`project = cen`. Selain itu, model otorisasi AWS tidak dapat cocok dengan kedua nilai. 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 placeholder yang dicetak miring dengan informasi akun Anda.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "TutorialAssumeSpecificRoles",
      "Effect": "Allow",
      "Action": "sts:AssumeRole",
      "Resource": [
        "arn:aws:iam::account-ID-without-hyphens:role/access-peg-
engineering",
        "arn:aws:iam::account-ID-without-hyphens:role/access-cen-
engineering"
      ]
    }
  ]
}
```

- Gunakan prosedur di [Langkah 4: Uji pembuatan rahasia](#) dan [Langkah 5: Uji menampilkan rahasia](#). Pada jendela peramban lain, konfirmasi 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. Konfirmasi 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

- 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 beralih peran di AWS Management Console, lihat [Beralih ke peran \(konsol\)](#).

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.

Rahasia ABAC memperbarui dan menghapus perilaku untuk setiap peran

Nama peran	Nama rahasia	Perilaku yang diharapkan
access-peg-engineering	test-access-peg-eng	Diizinkan
	test-access-uni-eng	Ditolak
	test-access-uni-qas	Ditolak
access-peg-quality-assurance	test-access-peg-qas	Diizinkan
	test-access-uni-eng	Ditolak
access-uni-engineering	test-access-uni-eng	Diizinkan
	test-access-uni-qas	Ditolak

Nama peran	Nama rahasia	Perilaku yang diharapkan
access-peg-quality-assurance	test-access-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. Akibatnya, Anda dapat masuk ke konsol IAM dengan peran pengujian Anda dan mengalami cara menggunakan tag untuk ABAC in. 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 memberikan izin administratif penuh kepada pengguna menggunakan kebijakan yang dikelola AdministratorAccess AWS, kebijakan ini tidak membatasi akses tersebut. Untuk informasi selengkapnya tentang bagaimana izin ditentukan saat beberapa kebijakan dilibatkan, lihat [Menentukan apakah permintaan diizinkan atau ditolak dalam sebuah akun..](#)

Sumber daya terkait

Untuk informasi terkait, lihat sumber daya berikut:

- [Untuk apa ABAC? AWS](#)
- [AWS kunci konteks kondisi global](#)
- [Membuat pengguna IAM \(konsol\)](#)
- [Membuat peran untuk mendelegasikan izin kepada pengguna IAM](#)
- [Menandai sumber daya IAM](#)

- [Mengontrol akses ke AWS sumber daya menggunakan tag](#)
- [Beralih ke peran \(konsol\)](#)
- [Tutorial IAM: Menggunakan tanda sesi SAML untuk ABAC](#)

Untuk mempelajari cara memantau tag di akun Anda, lihat [Memantau perubahan tag pada AWS sumber daya dengan alur kerja tanpa server dan Acara Amazon](#). CloudWatch

Tutorial IAM: Menggunakan tanda sesi SAML untuk ABAC

Kontrol akses berbasis atribut (ABAC) adalah strategi otorisasi yang mendefinisikan izin berdasarkan atribut. Di AWS, atribut ini disebut tanda. Anda dapat melampirkan tanda ke sumber daya IAM, termasuk entitas IAM (pengguna atau peran) dan keAWS sumber daya. Ketika entitas digunakan untuk membuat permintaan ke AWS, entitas tersebut menjadi prinsipal dan prinsipal tersebut mencakup tanda.

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 kebijakan AWS tradisional, yang mengharuskan Anda untuk mencantumkan setiap sumber daya individu. Untuk informasi selengkapnya tentang ABAC dan manfaatnya dibandingkan kebijakan tradisional, lihat [Untuk apa ABAC? AWS](#).

Jika perusahaan Anda menggunakan penyedia identitas (IdP) berbasis SAML untuk mengelola identitas pengguna perusahaan, Anda dapat menggunakan atribut SAML untuk kontrol akses yang lebih mendetail 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: Menentukan izin untuk mengakses sumber daya AWS berdasarkan tanda](#), 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 IdP berbasis SAML tempat Anda dapat membuat uji pengguna dengan atribut khusus.
- Kemampuan untuk masuk sebagai pengguna dengan izin administratif.
- Nikmati pengalaman membuat dan mengedit pengguna IAM, peran, dan kebijakan diAWS Management Console. Tetapi, jika Anda memerlukan bantuan untuk mengingat proses manajemen IAM, tutorial ABAC memberikan tautan yang bisa Anda gunakan untuk melihat step-by-step instruksi.
- Nikmati pengalaman menyiapkan IdP berbasis SAML di IAM. Untuk melihat perincian lebih lanjut dan tautan ke dokumentasi IAM terperinci, lihat [Melewati tag sesi menggunakan AssumeRoleWith SAMP](#).

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

Ketika Anda menggunakan tutorial ABAC untuk SAML, Anda harus melakukan langkah-langkah tambahan untuk membuat peran, mengonfigurasi IdP SAML dan mengaktifkan akses AWS Management Console. 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 peran berbasis SAML, lihat [Buat peran untuk federasi SAMP 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 [Mengubah peran \(konsol\)](#).

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.

```
{
  "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://signin.aws.amazon.com/saml"}
    }
  ]
}
```

Pernyataan `AllowSamlIdentityAssumeRole` memungkinkan anggota tim Teknik dan Jaminan Kualitas mengasumsikan peran ini ketika mereka digabungkan ke AWS dari IdP Example Corporation. Penyedia `ExampleCorpProvider` 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

Konfigurasi IdP SAML Anda untuk meneruskan atribut `cost-center`, `access-project`, dan `access-team` sebagai tanda sesi. Untuk informasi selengkapnya, lihat [Melewati tag sesi menggunakan AssumeRoleWith SAMP](#).

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 pengguna gabungan SAML 2.0 untuk mengakses AWS Management Console](#).

Langkah 4: Uji pembuatan rahasia

Gabungkan ke AWS Management Console menggunakan peran `access-session-tags`. Untuk informasi selengkapnya, lihat [Mengaktifkan pengguna gabungan SAML 2.0 untuk mengakses AWS Management Console](#). 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 IdP berbasis SAML Anda dengan atribut berikut ini:

- `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 harga di Secrets Manager, lihat [AWS Secrets Manager Penetapan 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 memberikan izin administratif penuh kepada pengguna menggunakan kebijakan yang dikelola AdministratorAccess AWS, kebijakan ini tidak membatasi akses tersebut. Untuk informasi selengkapnya tentang bagaimana izin ditentukan saat beberapa kebijakan dilibatkan, lihat [Menentukan apakah permintaan diizinkan atau ditolak dalam sebuah akun..](#)

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 AWS Management Console untuk mengonfigurasi kredensi (kunci akses, kata sandi, sertifikat penandatanganan, dan kunci publik SSH), menghapus atau menonaktifkan kredensial 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 kredensial 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:

Buat pengguna		Buat dan konfigurasi akun grup pengguna		
Nama pengguna	Instruksi lainnya	Nama grup pengguna	Tambahkan pengguna sebagai anggota	Instruksi lainnya
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.

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 AWS Management Console 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

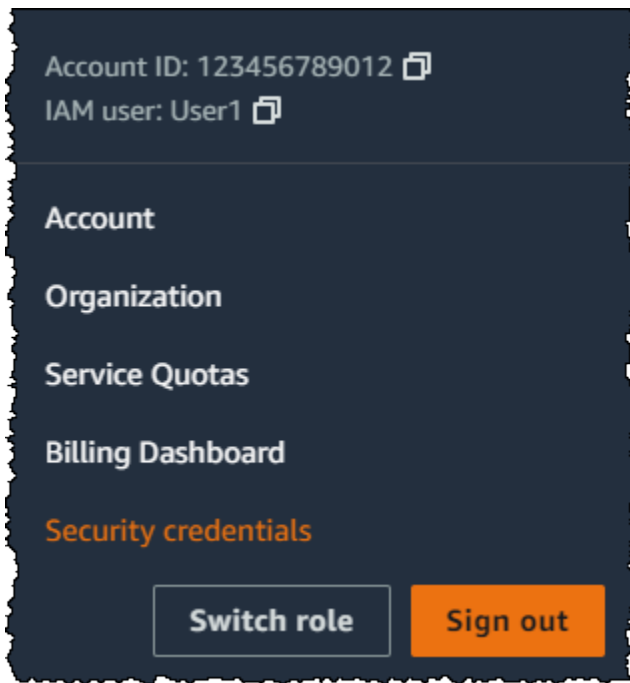
Selanjutnya Anda melampirkan dua kebijakan ke grup pengguna IAM pengujian, yang akan digunakan untuk memberikan izin yang dilindungi MFA.

1. Di panel navigasi, pilih User groups (Grup pengguna).
2. Pada 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 EC2MFA, di kotak pencarian, ketik **EC2Full**. Kemudian pilih kotak centang di sebelah AmazonEC2 FullAccess dalam daftar. Jangan menyimpan perubahan Anda.
5. Pada kotak pencarian, ketik **Force**, lalu pilih kotak centang di samping 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:

- [Menggunakan otentikasi multi-faktor \(MFA\) di AWS](#)
- [Mengaktifkan perangkat MFA untuk pengguna di AWS](#)
- [Menggunakan perangkat MFA dengan halaman masuk IAM Anda](#)

Identitas IAM (pengguna, pengguna, dan peran)

Tip

Mengalami masalah saat masuk AWS? Pastikan Anda berada di halaman login yang benar.

- Untuk masuk sebagai Pengguna root akun AWS (pemilik akun), gunakan kredensial yang Anda atur saat membuat. Akun AWS
- Untuk masuk sebagai pengguna IAM, gunakan kredensial yang diberikan administrator akun untuk masuk. AWS
- Untuk masuk dengan pengguna Pusat Identitas IAM, gunakan URL masuk yang dikirim ke alamat email Anda saat Anda membuat pengguna Pusat Identitas IAM.

Untuk bantuan masuk menggunakan pengguna Pusat Identitas IAM, lihat [Masuk ke portal AWS akses](#) di Panduan AWS Sign-In Pengguna.

Untuk tutorial masuk, lihat [Cara masuk di](#) Panduan AWS Sign-In Pengguna. AWS

Note

Jika Anda perlu meminta dukungan, jangan gunakan tautan Umpan Balik di halaman ini. Umpan balik yang Anda masukkan diterima oleh tim AWS Dokumentasi, bukan AWS Support. Sebagai gantinya, pilih tautan Hubungi Kami di bagian atas halaman ini. Di sana, Anda akan menemukan tautan ke sumber daya untuk membantu Anda mendapatkan dukungan yang Anda butuhkan.

Pengguna Pengguna root akun AWS atau administratif untuk akun dapat membuat identitas IAM. Identitas IAM menyediakan akses ke file Akun AWS. Grup pengguna IAM adalah kumpulan pengguna IAM yang dikelola sebagai satu unit. Identitas IAM mewakili pengguna manusia atau beban kerja terprogram, dan dapat diautentikasi dan kemudian diberi wewenang untuk melakukan tindakan. AWS Setiap identitas IAM dapat dikaitkan dengan satu kebijakan atau lebih. Kebijakan menentukan tindakan apa yang dapat dilakukan pengguna, peran, atau anggota grup pengguna, pada AWS sumber daya mana, dan dalam kondisi apa.

Akun AWS pengguna root

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 dan diakses dengan masuk dengan alamat email dan kata sandi yang Anda gunakan untuk membuat akun.

Important

Kami sangat menyarankan agar Anda tidak menggunakan pengguna root untuk tugas sehari-hari Anda. 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, lihat [Tugas yang memerlukan kredensial pengguna root](#).

Pengguna IAM

[Pengguna IAM](#) adalah identitas dalam diri Anda Akun AWS yang memiliki izin khusus untuk satu orang atau aplikasi. Jika memungkinkan, [praktik terbaik](#) merekomendasikan untuk mengandalkan kredensial sementara alih-alih membuat pengguna IAM yang memiliki kredensi jangka panjang seperti kata sandi dan kunci akses. Sebelum membuat kunci akses, tinjau [alternatif untuk kunci akses jangka panjang](#). Jika Anda memiliki kasus penggunaan khusus yang memerlukan kunci akses, kami sarankan Anda memperbarui kunci akses bila diperlukan. Untuk informasi selengkapnya, lihat [Perbarui kunci akses bila diperlukan untuk kasus penggunaan yang memerlukan kredensi jangka panjang](#). Untuk menambahkan pengguna IAM ke pengguna Anda Akun AWS, lihat [Membuat pengguna IAM di Akun AWS](#).

Note

Sebagai [praktik keamanan terbaik](#), kami menyarankan Anda menyediakan akses ke sumber daya Anda melalui federasi identitas alih-alih membuat pengguna IAM. Untuk informasi tentang situasi tertentu di mana pengguna IAM diperlukan, lihat [Kapan membuat pengguna IAM \(bukan peran\)](#).

Grup pengguna IAM

[Grup IAM](#) adalah identitas yang menentukan kumpulan pengguna IAM. Anda tidak dapat menggunakan grup untuk masuk. Anda dapat menggunakan grup untuk menentukan izin untuk beberapa pengguna sekaligus. Grup membuat izin lebih mudah dikelola untuk sekelompok besar pengguna. Misalnya, Anda dapat memiliki grup bernama IAMPublishers dan memberi grup itu jenis izin yang biasanya dibutuhkan oleh beban kerja penerbitan.

Peran IAM

[Peran IAM](#) adalah identitas dalam diri Anda Akun AWS yang memiliki izin khusus. Ini mirip dengan pengguna IAM, tetapi tidak terkait dengan orang tertentu. Anda dapat mengambil peran IAM untuk sementara AWS Management Console dengan [beralih peran](#). Anda dapat mengambil peran dengan memanggil operasi AWS CLI atau AWS API atau dengan menggunakan URL kustom. Untuk informasi selengkapnya tentang metode penggunaan peran, lihat [Menggunakan peran IAM](#).

Peran IAM dengan kredensi sementara digunakan dalam situasi berikut:

- Akses pengguna terfederasi – Untuk menetapkan izin ke identitas terfederasi, Anda membuat peran dan menentukan izin untuk peran tersebut. Saat identitas terfederasi mengautentikasi, identitas tersebut akan dikaitkan dengan peran dan diberi izin yang ditentukan oleh peran tersebut. Untuk informasi tentang peran-peran untuk federasi, lihat [Membuat peran untuk Penyedia Identitas pihak ketiga](#) dalam Panduan Pengguna IAM. Jika Anda menggunakan Pusat Identitas IAM, Anda perlu mengonfigurasi set izin. Untuk mengontrol apa yang dapat diakses identitas Anda setelah identitas tersebut diautentikasi, Pusat Identitas IAM mengorelasikan izin yang diatur ke peran dalam IAM. Untuk informasi tentang rangkaian izin, lihat [Rangkaian izin](#) dalam Panduan Pengguna AWS IAM Identity Center .
- Izin pengguna IAM sementara – Pengguna atau peran IAM dapat mengambil peran IAM guna mendapatkan berbagai izin secara sementara untuk tugas tertentu.
- Akses lintas akun – Anda dapat menggunakan peran IAM untuk mengizinkan seseorang (pengguna utama tepercaya) dengan akun berbeda untuk mengakses sumber daya yang ada di akun Anda. Peran adalah cara utama untuk memberikan akses lintas akun. Namun, pada beberapa Layanan AWS, Anda dapat melampirkan kebijakan secara langsung ke sumber daya (bukan menggunakan suatu peran sebagai perantara). Untuk mempelajari perbedaan antara peran dan kebijakan berbasis sumber daya untuk akses lintas akun, lihat [Akses sumber daya lintas akun di IAM](#)

- Akses lintas layanan — Beberapa Layanan AWS menggunakan fitur lain Layanan AWS. Sebagai contoh, ketika Anda melakukan panggilan dalam suatu layanan, biasanya layanan tersebut menjalankan aplikasi di Amazon EC2 atau menyimpan objek di Amazon S3. Sebuah layanan mungkin melakukannya menggunakan izin prinsipal yang memanggil, menggunakan peran layanan, atau peran terkait layanan.
- Sesi akses teruskan (FAS) — Saat Anda menggunakan pengguna atau peran IAM untuk melakukan tindakan AWS, Anda dianggap sebagai prinsipal. Saat Anda menggunakan beberapa layanan, Anda mungkin melakukan tindakan yang kemudian memulai tindakan lain di layanan yang berbeda. FAS menggunakan izin dari pemanggilan utama Layanan AWS, dikombinasikan dengan permintaan Layanan AWS untuk membuat permintaan ke layanan hilir. Permintaan FAS hanya dibuat ketika layanan menerima permintaan yang memerlukan interaksi dengan orang lain Layanan AWS atau sumber daya untuk menyelesaikannya. Dalam hal ini, Anda harus memiliki izin untuk melakukan kedua tindakan tersebut. Untuk detail kebijakan saat membuat permintaan FAS, lihat [Teruskan sesi akses](#).
- Peran layanan – Peran layanan adalah [peran IAM](#) yang diambil oleh layanan untuk melakukan tindakan atas nama Anda. Administrator IAM dapat membuat, memodifikasi, dan menghapus peran layanan dari dalam IAM. Untuk informasi selengkapnya, lihat [Membuat peran untuk mendelegasikan izin ke Layanan AWS](#) dalam Panduan pengguna IAM.
- Peran terkait layanan — Peran terkait layanan adalah jenis peran layanan yang ditautkan ke Layanan AWS. Layanan dapat menggunakan peran untuk melakukan tindakan atas nama Anda. Peran terkait layanan muncul di Akun AWS dan dimiliki oleh layanan. Administrator IAM dapat melihat, tetapi tidak dapat mengedit izin untuk peran terkait layanan.
- Aplikasi yang berjalan di Amazon EC2 — Anda dapat menggunakan peran IAM untuk mengelola kredensi sementara untuk aplikasi yang berjalan pada instans EC2 dan membuat atau permintaan API. AWS CLI AWS Cara ini lebih dianjurkan daripada menyimpan kunci akses dalam instans EC2. Untuk menetapkan AWS peran ke instans EC2 dan membuatnya tersedia untuk semua aplikasinya, Anda membuat profil instance yang dilampirkan ke instance. Profil instans berisi peran dan memungkinkan program yang berjalan di instans EC2 mendapatkan kredensial sementara. Untuk informasi selengkapnya, lihat [Menggunakan peran IAM untuk memberikan izin ke aplikasi yang berjalan di instans Amazon EC2](#) dalam Panduan Pengguna IAM.

[Kredensi sementara di IAM](#)

Sebagai [praktik terbaik](#), gunakan kredensial sementara dengan pengguna manusia dan beban kerja. Kredensial sementara terutama digunakan dalam peran IAM, tetapi juga ada penggunaan lain. Anda

dapat meminta kredensial sementara yang memiliki berbagai izin lebih terbatas dari pengguna IAM standar Anda. Ini mencegah Anda melakukan tugas secara tidak sengaja yang tidak diizinkan oleh kredensi yang lebih terbatas. Manfaat dari kredensial sementara adalah akan kedaluwarsa secara otomatis setelah jangka waktu tertentu. Anda memiliki kontrol selama kredensial tersebut valid.

Kapan menggunakan pengguna IAM Identity Center?

Kami menyarankan agar semua pengguna manusia menggunakan IAM Identity Center untuk mengakses AWS sumber daya. IAM Identity Center memungkinkan peningkatan signifikan dalam mengakses AWS sumber daya sebagai pengguna IAM. Pusat Identitas IAM menyediakan:

- Satu set identitas dan tugas sentral
- Akses ke akun di seluruh AWS Organisasi
- Koneksi ke penyedia identitas Anda yang ada
- Kredensial sementara
- Autentikasi multi-faktor (MFA)
- Konfigurasi MFA swalayan untuk pengguna akhir
- Penegakan administratif penggunaan MFA
- Single sign-on untuk semua hak Akun AWS

Untuk informasi selengkapnya, lihat [Apa itu Pusat Identitas IAM](#) di Panduan AWS IAM Identity Center Pengguna.

Kapan harus membuat pengguna IAM (alih-alih peran)

Kami menyarankan Anda hanya menggunakan pengguna IAM untuk kasus penggunaan yang tidak didukung oleh pengguna federasi. Beberapa kasus penggunaan meliputi:

- Beban kerja yang tidak dapat menggunakan peran IAM — Anda dapat menjalankan beban kerja dari lokasi yang perlu diakses. AWS Dalam beberapa situasi, Anda tidak dapat menggunakan peran IAM untuk memberikan kredensi sementara, seperti untuk plugin. WordPress Dalam situasi ini, gunakan kunci akses jangka panjang pengguna IAM agar beban kerja tersebut dapat diautentikasi. AWS
- AWS Klien pihak ketiga — Jika Anda menggunakan alat yang tidak mendukung akses dengan IAM Identity Center, 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 kredensi khusus layanan untuk CodeCommit mengautentikasi ke repositori Anda. Kami menyarankan Anda melakukan ini selain menggunakan pengguna di IAM Identity Center untuk otentikasi normal. Pengguna di IAM Identity Center adalah orang-orang di tenaga kerja Anda yang membutuhkan akses ke aplikasi cloud Anda Akun AWS atau ke aplikasi cloud Anda. Untuk memberi pengguna akses ke CodeCommit repositori Anda tanpa mengonfigurasi pengguna IAM, Anda dapat mengonfigurasi utilitas `git-remote-codecommit` Untuk informasi lebih lanjut tentang IAM dan CodeCommit, lihat [Menggunakan IAM dengan CodeCommit: Kredensial 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 Pengguna.AWS CodeCommit
- **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 kredensi khusus layanan untuk mengautentikasi dengan Amazon Keyspaces. Pengguna di IAM Identity Center adalah orang-orang di tenaga kerja Anda yang membutuhkan akses ke aplikasi cloud 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 Amazon Keyspaces 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 di dalamnya Akun AWS. Menetapkan akses darurat pengguna IAM dapat menjadi bagian dari rencana ketahanan Anda. Kami menyarankan agar kredensial pengguna darurat dikontrol dan diamankan dengan ketat menggunakan otentikasi multi-faktor (MFA).

Kapan harus membuat sebuah peran IAM (alih-alih pengguna)

Buat peran IAM dalam situasi berikut:

Anda membuat aplikasi yang berjalan pada instans Amazon Elastic Compute Cloud (Amazon EC2) dan aplikasi tersebut membuat permintaan. AWS

Jangan membuat pengguna IAM dan meneruskan kredensial pengguna ke aplikasi atau menanamkan kredensial di dalam aplikasi. Alih-alih, buat peran IAM yang Anda lampirkan ke instans EC2 untuk memberikan kredensial keamanan sementara pada aplikasi yang berjalan pada saat itu. Saat aplikasi menggunakan kredensial ini AWS, aplikasi dapat melakukan semua

operasi yang diizinkan oleh kebijakan yang dilampirkan pada peran tersebut. Untuk detailnya, lihat [Menggunakan peran IAM untuk memberikan izin pada aplikasi yang berjalan di instans Amazon EC2..](#)

Anda membuat aplikasi yang berjalan di ponsel dan aplikasi tersebut membuat permintaan ke AWS.

Jangan membuat pengguna IAM dan mendistribusikan kunci akses pengguna dengan aplikasi. Sebaliknya, gunakan penyedia seperti Login with Amazon, Amazon Cognito, Facebook, atau Google untuk mengautentikasi pengguna dan memetakan pengguna ke peran IAM. Aplikasi dapat menggunakan peran untuk mendapatkan kredensial keamanan sementara yang memiliki izin yang ditentukan oleh kebijakan yang terlampir pada peran tersebut. Untuk informasi selengkapnya, lihat hal berikut:

- [Panduan Pengguna Amazon Cognito](#)
- [Federasi OIDC](#)

Pengguna di perusahaan Anda diautentikasi di jaringan perusahaan Anda dan ingin dapat menggunakannya AWS tanpa harus masuk lagi-yaitu, Anda ingin mengizinkan pengguna untuk bergabung. AWS

Jangan buat pengguna IAM. Konfigurasi hubungan federasi antara sistem identitas perusahaan Anda dan AWS. Anda dapat melakukannya dengan dua cara:

- Jika sistem identitas perusahaan Anda kompatibel dengan SAMP 2.0, Anda dapat membangun kepercayaan antara sistem identitas perusahaan Anda dan AWS. Untuk informasi selengkapnya, lihat [Federasi SAML 2.0](#).
- Buat dan gunakan server proxy kustom yang menerjemahkan identitas pengguna dari perusahaan ke dalam peran IAM yang menyediakan kredensial keamanan sementara AWS. Untuk informasi selengkapnya, lihat [Mengaktifkan akses broker identitas khusus ke konsol AWS](#).

Bandungkan Pengguna root akun AWS kredensial dan kredensial pengguna IAM

Pengguna root adalah pemilik akun dan dibuat saat Akun AWS dibuat. Jenis pengguna lain, termasuk pengguna IAM, dan AWS IAM Identity Center pengguna dibuat oleh pengguna root atau administrator untuk akun tersebut. Semua AWS pengguna memiliki kredensial keamanan.

Kredensial pengguna root

Kredensial pemilik akun mengizinkan akses penuh ke semua sumber daya di akun. Anda tidak dapat menggunakan [kebijakan IAM](#) untuk secara eksplisit menolak akses pengguna root ke sumber daya. Anda hanya dapat menggunakan [kebijakan kontrol AWS Organizations layanan \(SCP\)](#) untuk membatasi izin pengguna root akun anggota. Karena itu, kami menyarankan Anda membuat pengguna administratif di IAM Identity Center untuk digunakan untuk AWS tugas sehari-hari. Kemudian, lindungi kredensi pengguna root dan gunakan untuk melakukan hanya beberapa tugas manajemen akun dan layanan yang mengharuskan Anda masuk sebagai pengguna root. Untuk daftar tugas-tugas tersebut, lihat [Tugas yang memerlukan kredensial pengguna root](#). Untuk mempelajari cara menyiapkan administrator untuk penggunaan sehari-hari di Pusat Identitas IAM, lihat [Memulai](#) di Panduan Pengguna Pusat Identitas IAM.

Kredensial IAM

Pengguna IAM adalah entitas yang Anda buat AWS yang mewakili orang atau layanan yang menggunakan pengguna IAM untuk berinteraksi dengan AWS sumber daya. Pengguna ini adalah identitas dalam diri Anda Akun AWS yang memiliki izin khusus khusus. Misalnya, Anda dapat membuat pengguna IAM dan memberi mereka izin untuk membuat direktori di IAM Identity Center. Pengguna IAM memiliki kredensi jangka panjang yang dapat mereka gunakan untuk mengakses AWS menggunakan AWS Management Console, atau secara terprogram menggunakan API atau AWS CLI AWS Untuk step-by-step petunjuk tentang cara pengguna IAM masuk AWS Management Console, lihat [Masuk ke pengguna IAM AWS Management Console sebagai pengguna IAM di Panduan Pengguna AWS](#) Masuk.

Secara umum, kami menyarankan Anda menghindari membuat pengguna IAM karena mereka memiliki kredensi jangka panjang seperti nama pengguna dan kata sandi. Sebagai gantinya, mewajibkan pengguna manusia untuk menggunakan kredensial sementara saat mengakses. AWS Anda dapat menggunakan penyedia identitas bagi pengguna manusia Anda untuk menyediakan akses federasi Akun AWS dengan mengasumsikan peran IAM, yang menyediakan kredensial sementara. Untuk manajemen akses terpusat, kami menyarankan Anda menggunakan [IAM Identity Center](#) untuk mengelola akses ke akun dan izin Anda 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 Pusat Identitas IAM di Panduan Pengguna Pusat](#) Identitas 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 dan diakses dengan masuk dengan alamat email dan kata sandi yang Anda gunakan untuk membuat akun.

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, lihat [Tugas yang memerlukan kredensial pengguna root](#).

Topik berikut mendetail tugas manajemen yang terkait dengan pengguna root.

Tugas

- [Aktifkan perangkat MFA virtual untuk Pengguna root akun AWS \(konsol\) Anda](#)
- [Aktifkan token TOTP perangkat keras untuk pengguna Akun AWS root \(konsol\)](#)
- [Aktifkan kunci keamanan FIDO untuk pengguna Akun AWS root \(konsol\)](#)
- [Ubah kata sandi untuk Pengguna root akun AWS](#)
- [Mengatur ulang kata sandi pengguna akar yang hilang atau terlupakan](#)
- [Membuat kunci akses untuk pengguna root](#)
- [Menghapus kunci akses untuk pengguna root](#)
- [Tugas yang memerlukan kredensial pengguna root](#)
- [Memecahkan masalah dengan pengguna root](#)
- [Informasi terkait](#)

Aktifkan perangkat MFA virtual untuk Pengguna root akun AWS (konsol) Anda

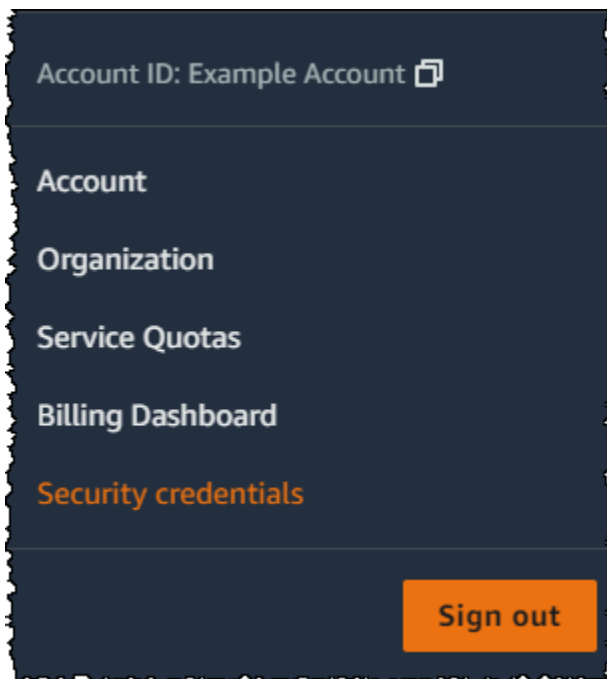
Anda dapat menggunakan AWS Management Console untuk mengkonfigurasi dan mengaktifkan perangkat MFA virtual untuk pengguna root Anda. Untuk mengaktifkan perangkat MFA Akun AWS, Anda harus masuk AWS menggunakan kredensial pengguna root Anda.

Sebelum Anda mengaktifkan MFA untuk pengguna akar Anda, tinjau pengaturan akun dan informasi kontak Anda 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 otentikasi alternatif ini, lihat [Bagaimana jika perangkat MFA hilang atau berhenti bekerja?](#).

Untuk mengonfigurasi dan mengaktifkan perangkat MFA virtual untuk digunakan dengan pengguna akar Anda (konsol)

1. Masuk ke AWS Management Console.
2. Di sisi kanan bilah navigasi, pilih nama akun Anda, dan pilih Kredensi keamanan. Jika perlu, pilih Continue to Security credentials.




3. Di bagian Multi-Factor Authentication (MFA), pilih Tetapkan perangkat MFA.
4. Di wizard, ketikkan nama Perangkat, pilih Aplikasi Authenticator, lalu pilih 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.

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 terikat dengan 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 Siapkan perangkat, pilih Tampilkan kunci rahasia, lalu ketik kunci rahasia ke dalam aplikasi MFA Anda.

 Important

Buat cadangan kode QR atau kunci konfigurasi rahasia yang aman, 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 menjadi 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 oleh [Pemulihan 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 kini siap digunakan dengan AWS. Untuk informasi lebih lanjut tentang penggunaan MFA dengan AWS Management Console, lihat [Menggunakan perangkat MFA dengan halaman masuk IAM Anda](#).

Aktifkan token TOTP perangkat keras untuk pengguna Akun AWS root (konsol)

Anda dapat mengonfigurasi dan mengaktifkan perangkat MFA fisik untuk pengguna root Anda dari AWS Management Console satu-satunya, bukan dari AWS CLI atau AWS API.

Jika perangkat MFA Anda hilang, dicuri, atau tidak berfungsi, Anda masih dapat masuk menggunakan faktor alternatif autentikasi. Jika Anda tidak dapat masuk dengan perangkat MFA Anda, Anda dapat masuk dengan memverifikasi identitas Anda menggunakan email dan telepon yang terdaftar dengan akun Anda. Sebelum Anda mengaktifkan MFA untuk pengguna akar Anda, tinjau pengaturan akun dan informasi kontak Anda untuk memastikan bahwa Anda memiliki akses ke email dan nomor telepon. Untuk mempelajari tentang proses masuk menggunakan faktor autentikasi alternatif, lihat [Bagaimana jika perangkat MFA hilang atau berhenti bekerja?](#). Untuk menonaktifkan fitur ini, hubungi [AWS Support](#).

ℹ Note

Anda mungkin melihat teks yang berbeda, seperti Masuk menggunakan MFA dan Atasi masalah perangkat autentikasi Anda. Namun, fitur yang sama disediakan. Dalam kasus lain, jika Anda tidak dapat memverifikasi alamat email akun dan nomor telepon Anda menggunakan faktor alternatif autentikasi, hubungi [AWS Support](#) untuk menonaktifkan pengaturan MFA Anda.

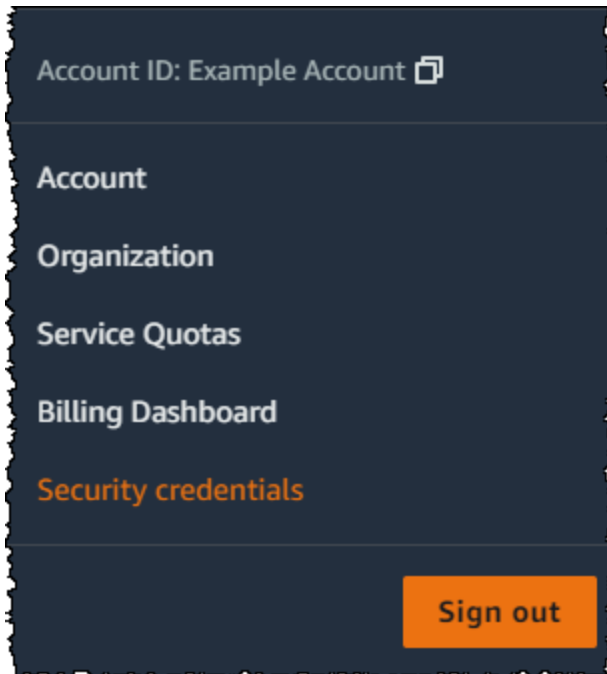
Untuk mengaktifkan perangkat MFA untuk pengguna akar Anda (konsol)

1. Masuk ke [Konsol IAM](#) sebagai pemilik akun dengan memilih Pengguna akar dan masukkan alamat email Akun AWS Anda. Di laman berikutnya, masukkan kata sandi Anda.

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 AWS Management Console 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. Perluas bagian Multi-Factor Authentication (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.

Aktifkan kunci keamanan FIDO untuk pengguna Akun AWS root (konsol)

Anda dapat mengonfigurasi dan mengaktifkan perangkat MFA virtual untuk pengguna root Anda dari AWS Management Console satu-satunya, bukan dari AWS CLI atau AWS API.

Jika kunci keamanan FIDO Anda hilang, dicuri, atau tidak berfungsi, Anda masih dapat masuk menggunakan perangkat MFA lain yang terdaftar di perangkat yang sama. Pengguna root akun AWS Jika Anda hanya memiliki satu perangkat MFA terdaftar, Anda dapat masuk menggunakan faktor identifikasi alternatif. Untuk mempelajari tentang proses masuk menggunakan faktor autentikasi alternatif, lihat [Bagaimana jika perangkat MFA hilang atau berhenti bekerja?](#). Untuk menonaktifkan fitur ini, hubungi [AWS Support](#).

Note

Anda tidak boleh memilih salah satu opsi yang tersedia di pop-up Google Chrome yang meminta Verifikasi identitas Anda dengan amazon.com. Anda hanya perlu menyetuk tombol keamanan.

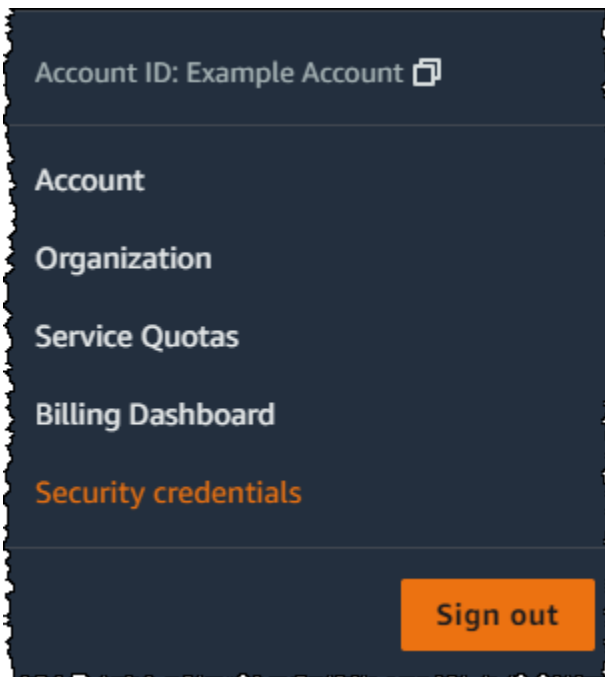
Untuk mengaktifkan kunci FIDO untuk pengguna root Anda (konsol)

1. Masuk ke [konsol IAM](#) sebagai pemilik akun dengan memilih pengguna Root dan memasukkan alamat Akun AWS email Anda. Di halaman berikutnya, masukkan kata sandi Anda.

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 AWS Management Console sebagai pengguna root](#) di PanduanAWS 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. Perluas bagian Multi-Factor Authentication (MFA).
4. Pilih Tetapkan perangkat MFA.
5. Di wizard, ketik nama Perangkat, pilih Kunci Keamanan, lalu pilih Berikutnya.
6. Masukkan kunci keamanan FIDO ke port USB komputer Anda.



7. Ketuk tombol keamanan FIDO.

Kunci keamanan FIDO siap digunakan. AWS Lain kali Anda menggunakan kredensi pengguna root Anda untuk masuk, Anda harus mengetuk kunci keamanan FIDO Anda untuk menyelesaikan proses masuk.

Untuk membantu memecahkan masalah dengan kunci keamanan FIDO Anda, lihat [Mengonfigurasi tombol keamanan FIDO](#)

Ubah kata sandi untuk Pengguna root akun AWS

Anda dapat mengubah alamat email dan kata sandi dari [Kredensial Keamanan](#) atau halaman Akun. Anda juga dapat memilih Lupa kata sandi? pada halaman AWS login 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 [Mengatur ulang kata sandi pengguna akar 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 mengetahui 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.

AWS Management Console

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. Gunakan alamat email dan kata sandi Anda untuk masuk ke [AWS Management Console](#) sebagai Akun AWS milik Anda Pengguna root akun AWS.
2. Di sudut kanan atas konsol, pilih nama atau nomor akun Anda, lalu pilih Akun.
3. Pada halaman Akun, di samping Pengaturan akun, pilih Edit. Anda diminta untuk mengautentikasi 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 Perbarui pengaturan akun, di bawah Kata Sandi, pilih Edit.
5. Pada halaman Perbarui kata sandi Anda, isi kolom untuk Kata sandi saat ini, Kata sandi baru, dan Konfirmasi kata sandi baru.

⚠ Important

Pastikan untuk memilih kata sandi 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 ketentuan berikut:

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

ℹ Note

AWS sedang meluncurkan peningkatan pada proses masuk. Salah satu peningkatan tersebut adalah menerapkan kebijakan kata sandi yang lebih aman untuk akun Anda. Jika AWS telah meningkatkan akun Anda, Anda harus memenuhi kebijakan kata sandi yang dijelaskan sebelumnya. Jika AWS belum memutakhirkan akun Anda, maka AWS belum menerapkan kebijakan ini. Namun, kami sangat menyarankan agar Anda mengikuti pedomannya untuk kata sandi yang lebih aman.

6. Pilih Simpan perubahan.

AWS CLI or AWS SDK


Tugas ini tidak didukung di AWS CLI atau oleh operasi API dari salah satu AWS SDK. Anda dapat melakukan tugas ini hanya dengan menggunakan AWS Management Console.

Mengatur ulang kata sandi pengguna akar yang hilang atau terlupakan

Saat pertama kali membuat Akun AWS, Anda memberikan alamat email dan kata sandi. Ini adalah Pengguna root akun AWS kredensialmu. Jika Anda lupa kata sandi pengguna root Anda, Anda dapat mengatur ulang kata sandi dari file AWS Management Console.


Untuk mengatur ulang kata sandi pengguna akar Anda:

1. Gunakan alamat Akun AWS email Anda untuk mulai masuk ke [AWS Management Console](#) sebagai pengguna root dan kemudian pilih Berikutnya.

 Note

Jika Anda masuk ke kredensi pengguna IAM [AWS Management Console](#) with, maka Anda harus keluar sebelum dapat mengatur ulang kata sandi pengguna root. Jika Anda melihat halaman masuk pengguna IAM khusus akun, pilih Masuk menggunakan kredensial akun akar 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 adalah pengguna IAM, opsi ini tidak tersedia. Lupa 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 akun saya AWS](#). Jika Anda masuk melalui Portal akses AWS, lihat [Menyetel 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 berakhiran@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.

Membuat kunci akses untuk pengguna root

Warning

Kami sangat menyarankan Anda untuk tidak membuat access key pair untuk pengguna root Anda. Karena [hanya beberapa tugas yang memerlukan pengguna root](#) dan Anda biasanya jarang melakukan tugas-tugas tersebut, kami sarankan masuk ke AWS Management Console 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 Anda 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 access key, AWS memberi Anda satu kesempatan untuk melihat dan mengunduh bagian secret access key dari access key. Jika Anda tidak mengunduhnya atau Anda kehilangannya, Anda dapat menghapus access key itu lalu membuat yang baru. Anda dapat membuat root access key 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 access key ke pengguna root.

Kunci akses yang tidak digunakan harus dinonaktifkan. Setelah access key tidak aktif, Anda tidak dapat menggunakannya untuk panggilan API. Kunci tidak aktif masih dihitung terhadap 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.

AWS Management Console

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. Gunakan alamat email dan kata sandi Anda untuk masuk ke [Memulai dengan AWS Management Console](#) sebagai Akun AWS milik Anda. Pengguna root akun AWS.
2. Di sudut kanan atas konsol, pilih nama atau nomor akun Anda dan kemudian pilih Security Credentials.
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 access key yang ada sebelum Anda dapat membuat kunci baru. Untuk informasi lebih lanjut, lihat [IAM Object Quotas](#).
4. Pada halaman Alternatives to root user access keys, tinjau rekomendasi keamanan. Untuk melanjutkan, pilih kotak centang, dan kemudian pilih Buat access key.
5. Pada halaman Retrieve access key, ID kunci Access Anda akan ditampilkan.
6. Di bawah Secret access key, pilih Tampilkan dan kemudian menyalin access key ID dan kunci rahasia dari jendela browser Anda dan menaruhkannya di tempat yang aman. Atau, Anda dapat memilih Download file csv yang akan men-download file bernama `rootkey.csv` yang berisi access key ID dan kunci rahasia. Simpan file di tempat yang aman.
7. Pilih Selesai. Ketika Anda tidak lagi membutuhkan access key, [kami sarankan Anda menghapusnya](#), atau setidaknya mempertimbangkan untuk menonaktifkannya sehingga tidak ada yang dapat menyalahgunakannya.

AWS CLI & SDKs

Untuk membuat access key untuk pengguna root

Note

Untuk menjalankan perintah berikut atau operasi API sebagai pengguna root, Anda harus sudah memiliki satu active access key pair. Jika Anda tidak memiliki kunci akses, buat kunci akses pertama menggunakan tombol AWS Management Console. Kemudian, Anda dapat menggunakan kredensial dari kunci akses pertama dengan AWS CLI untuk membuat kunci akses kedua, atau untuk menghapus kunci akses.

- AWS CLI: [aws iam create-access-key](#)

Example

```
$ aws iam create-access-key
{
  "AccessKey": {
    "UserName": "MyUserName",
    "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "Status": "Active",
    "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY",
    "CreateDate": "2021-04-08T19:30:16+00:00"
  }
}
```

- AWSAPI: [CreateAccessKey](#) dalam Referensi API IAM.

Menghapus kunci akses untuk pengguna root

Anda dapat menggunakan AWS Management Console, AWS CLI atau AWS API untuk menghapus kunci akses pengguna root.

AWS Management Console

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. Gunakan alamat email dan kata sandi Anda untuk masuk ke [Memulai dengan AWS Management Console](#) sebagai Akun AWS milik Anda. Pengguna root akun AWS.
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

Important

Mengalami masalah saat 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 kredensial yang Anda atur saat membuat Akun AWS. Jika Anda adalah pengguna IAM, administrator akun Anda dapat memberi Anda kredensial yang dapat Anda gunakan untuk masuk. AWS. Jika Anda perlu meminta dukungan, jangan gunakan tautan umpan balik di halaman ini, karena formulir diterima oleh tim AWS Dokumentasi, bukan AWS Support. Sebagai gantinya, pada halaman [Hubungi Kami](#) pilih Masih tidak dapat masuk ke AWS akun Anda dan kemudian pilih salah satu opsi dukungan yang tersedia.

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.

Tugas Manajemen Akun

- [Ubah pengaturan akun Anda](#). Ini mencakup nama akun, alamat email, kata sandi pengguna root, dan access key pengguna root. Pengaturan akun lainnya, seperti informasi kontak, preferensi mata uang pembayaran, dan Wilayah AWS, tidak memerlukan kredensial pengguna root.
- [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.
- [Tutup Anda Akun AWS](#).

Untuk informasi selengkapnya, lihat topik berikut:

- [Bagaimana cara menetapkan kepemilikan saya Akun AWS ke entitas lain?](#) .
- [Bagaimana cara menutup saya Akun AWS?](#) .
- [Tutup mandiri Akun AWS](#).

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 untuk AWS GovCloud \(US\)](#).
- Minta kunci akses pengguna root AWS GovCloud (US) akun dari AWS Support.
- Jika AWS Key Management Service kunci menjadi tidak dapat dikelola, Anda dapat memulihkannya dengan menghubungi AWS Support sebagai pengguna root.

Tugas Amazon EC2

- [Mendaftar sebagai penjual](#) di Marketplace Instans Cadangan.

Tugas Layanan Penyimpanan Sederhana Amazon

- [Konfigurasi bucket Amazon S3 untuk mengaktifkan MFA \(otentikasi multi-faktor\)](#).
- [Edit atau hapus kebijakan bucket Amazon S3 yang menyangkal semua prinsipal](#).

Tugas Layanan Antrian Sederhana Amazon

- [Mengedit atau menghapus kebijakan sumber daya Amazon SQS yang menyangkal semua prinsipal](#).

Memecahkan masalah dengan pengguna root

Gunakan informasi di sini untuk membantu Anda memecahkan masalah yang terkait dengan pengguna root. Akun AWS

Saya tidak dapat melakukan tugas yang saya harapkan dapat dilakukan saat masuk sebagai pengguna root akun

Jika Anda tidak dapat menyelesaikan tugas saat masuk sebagai pengguna root untuk akun tersebut, akun Anda mungkin merupakan anggota organisasi yang masuk AWS Organizations. Jika demikian, dan administrator organisasi Anda menggunakan kebijakan kontrol layanan (SCP) untuk membatasi izin akun Anda, maka semua pengguna, termasuk pengguna root, terpengaruh. Untuk informasi selengkapnya, lihat [Kebijakan kontrol layanan](#) di 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 [Mengatur ulang kata sandi pengguna akar 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 mengetahui alamat email tetapi tidak lagi memiliki akses ke email, coba pulihkan akses ke email terlebih dahulu menggunakan salah satu opsi berikut:

- Jika Anda memiliki domain untuk alamat email, Anda dapat mengembalikan alamat email yang dihapus. Atau, Anda dapat menyiapkan catch-all untuk akun email Anda, yang mengirimkan pesan “menangkap semua” 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](#).

Informasi terkait

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 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.

Pengguna AWS Identity and Access Management (IAM) adalah entitas yang Anda buat. AWS Pengguna IAM mewakili pengguna manusia atau beban kerja yang menggunakan pengguna IAM untuk berinteraksi dengannya. AWS Pengguna di AWS terdiri atas 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 menciptakan cara-cara berikut untuk mengidentifikasi pengguna tersebut:

- Sebuah “nama ramah” untuk pengguna IAM, yang merupakan nama yang Anda tentukan ketika Anda membuat pengguna IAM, seperti Richard atau. Anaya Ini adalah nama yang Anda lihat di AWS Management Console.
- Nama Sumber Daya Amazon (ARN) untuk pengguna IAM. Anda menggunakan ARN ketika Anda perlu mengidentifikasi pengguna IAM secara unik di semua. AWS Misalnya, Anda dapat menggunakan ARN untuk menentukan pengguna IAM sebagai Principal 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 ketika Anda menggunakan API, Alat untuk 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 kredensialnya IAM

Anda dapat mengakses dengan berbagai AWS cara tergantung pada kredensi pengguna IAM:

- [Kata sandi konsol](#): Kata sandi yang dapat diketik pengguna IAM untuk masuk ke sesi interaktif seperti. AWS Management Console Menonaktifkan kata sandi (akses konsol) untuk pengguna IAM mencegah mereka masuk ke AWS Management Console menggunakan kredensial masuk mereka. Hal ini tidak mengubah izin mereka atau mencegah mereka mengakses konsol menggunakan peran yang diasumsikan.
- [Kunci 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, AWS API PowerShell, atau AWS Console Mobile Application.
- [Kunci SSH untuk digunakan dengan CodeCommit](#): Kunci publik SSH dalam format OpenSSH yang dapat digunakan untuk diautentikasi dengan CodeCommit.
- [Sertifikat server](#): Sertifikat SSL/TLS yang dapat Anda gunakan untuk mengautentikasi IT beberapa layanan AWS. 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. Ketika Anda menggunakan AWS Management Console 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 kredensi apa pun. Anda harus membuat jenis kredensi 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 AWS Management Console. Atur kebijakan kata sandi untuk menerapkan kerumitan kata sandi minimum. Izinkan pengguna 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 harus memberikan dua bentuk identifikasi: Pertama, mereka memberikan kredensi 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 pada 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 untuk menghapus kata sandi dan access key saat pengguna tidak lagi membutuhkannya.
- [Unduh laporan kredensial 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 sebagian besar kasus, Anda ingin membatasi izin pengguna untuk hanya tugas (tindakan atau operasi AWS) dan sumber daya yang diperlukan untuk tugas.

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 dan memberi mereka kredensial dan izin awal, lihat [Membuat 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 [Mengelola kata sandi pengguna di AWS](#) dan [Mengelola kunci akses 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 oleh kebijakan berbasis identitas kepada pengguna atau peran IAM. Untuk informasi selengkapnya tentang tipe dan penggunaan kebijakan, lihat [Kebijakan dan Izin di IAM](#).

Pengguna dan akun IAM

Setiap pengguna IAM dikaitkan dengan satu dan hanya satu Akun AWS. Karena pengguna IAM didefinisikan dalam Akun AWS, mereka tidak perlu memiliki metode pembayaran yang tersimpan. 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 permintaan. AWS Ini biasanya disebut sebagai akun layanan. Jika Anda memilih untuk menggunakan kredensial jangka panjang seorang pengguna IAM di aplikasi Anda, jangan menyimpan kunci akses secara langsung ke kode aplikasi Anda. SDK AWS dan AWS Command Line Interface memungkinkan Anda menempatkan access key di lokasi yang diketahui sehingga Anda tidak perlu menyimpannya dalam kode. Untuk informasi selengkapnya, lihat [Mengelola Kunci Akses Pengguna IAM dengan Benar](#) di Referensi Umum AWS Atau, sebagai praktik terbaik, Anda dapat [menggunakan kredensial keamanan sementara \(peran IAM\) alih-alih kunci akses jangka panjang](#).

Membuat pengguna IAM di Akun AWS

 [Follow us on Twitter](#)

⚠ Important

IAM [praktik terbaik](#) menyarankan agar Anda meminta pengguna manusia untuk menggunakan federasi dengan penyedia identitas untuk mengakses AWS menggunakan kredensi sementara alih-alih menggunakan pengguna IAM dengan kredensi jangka panjang.

ℹ Note

Jika Anda menemukan halaman ini karena Anda mencari informasi tentang API Iklan Produk untuk menjual produk Amazon di situs web Anda, lihat [Dokumentasi Iklan Produk API 5.0](#). Jika Anda sampai di halaman ini dari konsol IAM, kemungkinan akun Anda tidak menyertakan pengguna IAM, meskipun Anda sudah masuk. Anda dapat masuk sebagai Pengguna root akun AWS, menggunakan peran, atau masuk menggunakan kredensial sementara. Untuk mempelajari lebih banyak tentang identitas IAM ini, lihat [Identitas IAM \(pengguna, pengguna, dan peran\)](#).

Proses membuat pengguna dan memungkinkan pengguna tersebut melakukan tugas kerja terdiri atas langkah-langkah berikut:

1. Buat pengguna di AWS Management Console, yang AWS CLI, Alat untuk Windows PowerShell, atau menggunakan AWS Operasi API. Jika Anda membuat pengguna di AWS Management Console, kemudian langkah 1—4 ditangani secara otomatis, berdasarkan pilihan Anda. Jika Anda membuat pengguna secara terprogram, Anda harus melakukan setiap langkah tersebut secara terpisah.
2. Buat kredensial untuk pengguna, tergantung pada jenis akses yang diperlukan pengguna:
 - Aktifkan akses konsol —pilihan: Jika pengguna perlu mengakses AWS Management Console, [buat kata sandi untuk pengguna](#). Menonaktifkan akses konsol untuk pengguna mencegah mereka masuk ke AWS Management Console 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 kredensi yang dibutuhkan pengguna. Misalnya, untuk pengguna yang membutuhkan akses hanya melalui AWS Management Console, jangan membuat access key.

3. Berikan izin pengguna untuk melakukan tugas yang diperlukan dengan menambahkan pengguna ke satu atau beberapa grup. Anda juga dapat memberikan izin dengan melampirkan kebijakan izin secara langsung kepada pengguna. Namun, kami menyarankan agar Anda menempatkan pengguna Anda dalam grup dan mengelola izin melalui kebijakan yang dilampirkan ke grup tersebut. Anda juga dapat menggunakan [batas izin](#) untuk membatasi izin yang dapat dimiliki pengguna, meskipun ini bukan hal yang umum.
4. (Opsional) Tambahkan metadata ke pengguna dengan melampirkan tag. Untuk informasi selengkapnya tentang menggunakan tag di IAM, lihat [Menandai sumber daya IAM](#).
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 M M M IAM IAM IAM IAM IMAWS](#).
6. (Opsional) Konfigurasi [Multi-Factor Authentication \(MFA\)](#) bagi pengguna. MFA mengharuskan pengguna untuk memberikan one-time-usekode setiap kali dia menandatanganiAWS Management Console.
7. (Opsional) Berikan izin kepada pengguna untuk mengelola kredensial keamanan mereka sendiri. (Secara default, pengguna tidak memiliki izin untuk mengelola kredensial mereka sendiri.) Untuk informasi selengkapnya, lihat [Mengizinkan pengguna IAM untuk mengubah kata sandi mereka sendiri](#).

Untuk informasi tentang izin yang Anda butuhkan untuk membuat pengguna, lihat [Izin diperlukan untuk mengakses sumber daya IAM](#).

Topik

- [Membuat pengguna IAM \(konsol\)](#)
- [Membuat pengguna IAM \(AWS CLI\)](#)
- [Membuat pengguna IAM \(AWSAPI\)](#)

Membuat pengguna IAM (konsol)

Anda dapat menggunakan AWS Management Console untuk membuat pengguna IAM.

Untuk membuat pengguna IAM (konsol)

1. Ikuti prosedur login yang sesuai dengan jenis pengguna Anda seperti yang dijelaskan dalam topik [Cara masuk ke AWS](#) di dalam [AWS Panduan Pengguna Masuk](#).
2. Pada [Rumah Konsol](#) halaman, pilih layanan IAM.
3. Di panel navigasi, pilih [Penggunaan](#) dan kemudian pilih [Tambahkan pengguna](#).
4. Pada [Tentukan detail pengguna](#) halaman, di bawah [Detail pengguna](#), di [Nama pengguna](#), masukkan nama untuk pengguna baru. Ini adalah nama masuk mereka untuk AWS.

Note

Jumlah dan ukuran sumber daya IAM dalam AWS akun terbatas. Untuk informasi selengkapnya, lihat [IAM dan AWS STS kuota](#). Nama pengguna dapat berupa kombinasi hingga 64 huruf, digit, dan karakter ini: plus (+), sama dengan (=), koma (,), titik (.), pada a keong (@), garis bawah (_), dan tanda hubung (-). Nama harus unik dalam akun. Grup 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 adalah case sensitive. Ketika nama pengguna muncul kepada pelanggan di konsol, seperti selama proses login, nama pengguna tidak peka huruf besar.

5. Pilih [Menyediakan akses pengguna ke -AWS Management Console pilihan](#) ini menghasilkan [AWS Management Console kredensi login](#) untuk pengguna baru.


Anda ditanya apakah Anda menyediakan akses konsol ke seseorang. Kami menyarankan Anda membuat pengguna di IAM Identity Center daripada IAM.

- Untuk beralih ke membuat pengguna di IAM Identity Center, pilih [Tentukan pengguna di Pusat Identitas](#).

Jika Anda belum mengaktifkan IAM Identity Center, memilih opsi ini akan membawa Anda ke halaman layanan di konsol sehingga Anda dapat mengaktifkan layanan. Untuk detail tentang prosedur ini, lihat <https://docs.aws.amazon.com/singlesignon/latest/userguide/getting-started.html> di dalam [AWS IAM Identity Center Panduan Pengguna](#)

Jika Anda telah mengaktifkan IAM Identity Center, memilih opsi ini akan membawa Anda ke Tentukan detail penggunahalaman di IAM Identity Center. Untuk detail tentang prosedur ini, lihat <https://docs.aws.amazon.com/singlesignon/latest/userguide/addusers.html> di dalam AWS IAM Identity Center Panduan Pengguna

- Jika Anda tidak dapat menggunakan IAM Identity Center, pilih Saya ingin membuat pengguna IAM dan terus mengikuti prosedur ini.
- a. Untuk Kata sandi konsol, pilih salah satu dari berikut ini:
 - Sandi yang dibuat secara otomatis- Pengguna mendapat kata sandi yang dibuat secara acak yang memenuhi [kebijakan kata sandi akun](#). Anda dapat melihat atau mengunduh kata sandi saat Anda sampai di Ambil kata sandi halaman.
 - Kata sandi khusus- Pengguna diberi kata sandi yang Anda masukkan di dalam kotak.
- b. (Opsional) Pengguna harus membuat kata sandi baru pada saat login berikutnya (disarankan) dipilih secara default untuk memastikan bahwa pengguna dipaksa untuk mengubah kata sandi mereka saat pertama kali masuk.

 Note

Jika administrator telah mengaktifkan [Izinkan pengguna untuk mengubah kata sandi mereka sendiri](#) pengaturan kebijakan kata sandi akun, maka kotak centang ini tidak melakukan apa-apa. Jika tidak, akan otomatis melampirkan kebijakan terkelola AWS yang bernama [IAMUserChangePassword](#) ke pengguna baru. Kebijakan tersebut memberi mereka izin untuk mengubah kata sandi mereka sendiri.

6. Pilih Selanjutnya.
7. Pada Tetapkan izin halaman, tentukan bagaimana Anda ingin menetapkan izin untuk pengguna ini. Pilih salah satu dari tiga opsi berikut:
 - Tambahkan pengguna ke grup- Pilih opsi ini jika Anda ingin menetapkan pengguna ke satu atau beberapa grup yang sudah memiliki kebijakan izin. IAM menampilkan daftar kelompok di akun Anda, di sepanjang kebijakan terlampir mereka. Anda dapat memilih satu atau beberapa grup yang ada, atau memilih Buat grup untuk membuat grup baru. Untuk informasi selengkapnya, lihat [Mengubah izin untuk pengguna IAM](#).
 - Salin izin— Pilih opsi ini untuk menyalin semua keanggotaan grup, kebijakan terkelola terlampir, kebijakan sebaris tertanam, dan semua yang ada [batas izin](#) dari pengguna yang

sudah ada ke pengguna baru. IAM menampilkan daftar pengguna di akun Anda. Pilih salah satu yang izinnya paling sesuai dengan kebutuhan pengguna baru Anda.

- Lampirkan kebijakan secara langsung- Pilih opsi ini untuk melihat daftarAWSkebijakan yang dikelola dan dikelola pelanggan di akun Anda. Pilih kebijakan yang ingin Anda lampirkan ke pengguna atau pilihBuat kebijakanuntuk membuka tab browser baru dan membuat kebijakan baru. Untuk informasi selengkapnya, lihat langkah 4 di prosedur [Membuat kebijakan IAM](#) . Setelah Anda membuat kebijakan, tutup tab itu dan kembali ke tab asli untuk menambahkan kebijakan ke pengguna.

 Tip

Bila memungkinkan, lampirkan kebijakan Anda ke grup dan kemudian jadikan pengguna anggota grup yang sesuai.

8. (Opsional) Tetapkan [batas izin](#). Ini adalah fitur lanjutan.

BukaBatas izinbagian dan pilihGunakan batas izin untuk mengontrol izin maksimum. IAM menampilkan daftarAWSkebijakan yang dikelola dan dikelola pelanggan di akun Anda. Pilih kebijakan yang akan digunakan untuk batas izin atau pilihBuat kebijakanuntuk membuka tab browser baru dan membuat kebijakan baru. Untuk informasi selengkapnya, lihat langkah 4 dalam prosedur [Membuat kebijakan IAM](#) . Setelah Anda membuat kebijakan, tutup tab tersebut dan kembali ke tab asli Anda untuk memilih kebijakan yang akan digunakan untuk batas izin.

9. Pilih Selanjutnya.
10. (Opsional) PadaTinjau dan buathalaman, di bawahTag, pilihTambahkan tag baruuntuk menambahkan metadata ke pengguna dengan melampirkan tag sebagai pasangan kunci-nilai. Untuk informasi selengkapnya tentang menggunakan tanda di IAM, lihat [Menandai sumber daya IAM](#).
11. Tinjau semua pilihan yang Anda buat sampai saat ini. Ketika Anda siap untuk melanjutkan, pilihBuat pengguna.
12. PadaAmbil kata sandihalaman, mendapatkan password yang ditugaskan untuk pengguna:
 - PilihTampilkandi samping kata sandi untuk melihat kata sandi pengguna sehingga Anda dapat merekamnya secara manual.
 - PilihUnduh .csvuntuk mengunduh kredensi login pengguna sebagai file.csv yang dapat Anda simpan ke lokasi yang aman.

13. Pilih Petunjuk masuk email. Klien email lokal Anda terbuka dengan draf yang dapat Anda sesuaikan dan kirim ke pengguna. Templat email mencakup perincian-perincian berikut untuk setiap pengguna:
 - Nama pengguna
 - URL ke halaman masuk akun. Gunakan contoh berikut, yang mengganti nomor ID akun atau alias akun yang benar:

```
https://AWS-account-ID or alias.signin.aws.amazon.com/console
```

Important

Kata sandi pengguna tidak disertakan dalam email yang dibuat. Anda harus memberikan kata sandi kepada pengguna dengan cara yang sesuai dengan pedoman keamanan organisasi Anda.

14. Jika pengguna juga membutuhkan kunci akses, lihat [Mengelola kunci akses untuk pengguna IAM](#).

Membuat pengguna IAM (AWS CLI)

Anda dapat menggunakan AWS CLI untuk membuat pengguna IAM.

Untuk membuat pengguna IAM (AWS CLI)

1. Buat pengguna.
 - [aws iam membuat-pengguna](#)
2. (Opsional) Berikan akses pengguna ke AWS Management Console. Ini memerlukan kata sandi. Anda juga harus memberikan [URL halaman masuk akun Anda](#) kepada pengguna.
 - [AWScreate-login-profile](#)
3. (Opsional) Berikan akses terprogram pengguna. Hal ini memerlukan access key.
 - [AWScreate-access-key](#)
 - Alat untuk Windows PowerShell: [iam baruAccessKey](#)
 - API IAM: [CreateAccessKey](#)

⚠ 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.

4. Tambahkan pengguna ke satu atau lebih grup. Grup yang Anda sebutkan harus memiliki kebijakan terlampir yang memberikan izin yang sesuai untuk pengguna.
 - [AWSadd-user-to-group](#)
5. (Opsional) Lampirkan kebijakan ke pengguna yang menentukan izin pengguna. Catatan: Sebaiknya Anda mengelola izin pengguna dengan menambahkan pengguna ke grup dan melampirkan kebijakan ke grup, alih-alih langsung ke pengguna.
 - [AWSattach-user-policy](#)
6. (Opsional) Tambahkan atribut khusus ke pengguna dengan melampirkan tag. Untuk informasi selengkapnya, lihat [Mengelola tag pada pengguna IAM \(AWS CLI atau AWS API\)](#).
7. (Opsional) Berikan izin kepada pengguna untuk mengelola kredensial keamanan mereka sendiri. Untuk informasi selengkapnya, lihat [AWS: Memungkinkan pengguna IAM yang diautentikasi MFA untuk mengelola kredensialnya sendiri di halaman kredensi Keamanan](#).

Membuat pengguna IAM (AWS API)

Anda dapat menggunakan AWS API untuk membuat pengguna IAM.

Untuk membuat pengguna IAM dari (AWS API)

1. Buat pengguna.
 - [CreateUser](#)
2. (Opsional) Berikan akses pengguna ke AWS Management Console. Ini memerlukan kata sandi. Anda juga harus memberikan [URL halaman masuk akun Anda](#) kepada pengguna.
 - [CreateLoginProfile](#)
3. (Opsional) Berikan akses terprogram pengguna. Hal ini memerlukan access key.

- [CreateAccessKey](#)

 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.

4. Tambahkan pengguna ke satu atau lebih grup. Grup yang Anda sebutkan harus memiliki kebijakan terlampir yang memberikan izin yang sesuai untuk pengguna.

- [AddUserToGroup](#)

5. (Opsional) Lampirkan kebijakan ke pengguna yang menentukan izin pengguna. Catatan: Sebaiknya Anda mengelola izin pengguna dengan menambahkan pengguna ke grup dan melampirkan kebijakan ke grup, alih-alih langsung ke pengguna.


- [AttachUserPolicy](#)

6. (Opsional) Tambahkan atribut khusus ke pengguna dengan melampirkan tag. Untuk informasi selengkapnya, lihat [Mengelola tag pada pengguna IAM \(AWS CLI atau AWS API\)](#).

7. (Opsional) Berikan izin kepada pengguna untuk mengelola kredensial keamanan mereka sendiri. Untuk informasi selengkapnya, lihat [AWS: Memungkinkan pengguna IAM yang diautentikasi MFA untuk mengelola kredensialnya sendiri di halaman kredensi Keamanan](#).

Mengontrol akses pengguna IAM ke AWS Management Console

Pengguna IAM dengan izin yang masuk ke Akun AWS melalui AWS Management Console dapat mengakses AWS sumber daya IAM Anda. Daftar berikut menunjukkan cara yang dapat Anda gunakan untuk memberi pengguna IAM akses ke Akun AWS sumber daya IAM Anda melalui AWS Management Console. Juga menunjukkan bagaimana pengguna IAM dapat mengakses fitur AWS akun lainnya melalui situs AWS web.

 Note

Tidak ada biaya untuk menggunakan IAM.

Yang AWS Management Console

Anda membuat kata sandi untuk setiap pengguna IAM yang memerlukan akses ke AWS Management Console. Pengguna mengakses konsol melalui halaman Akun AWS masuk dengan IAM yang diaktifkan. Untuk informasi tentang mengakses halaman login, lihat [Cara masuk ke AWS dalam](#) Panduan AWS Sign-In Pengguna. Untuk informasi tentang cara membuat kata sandi, lihat [Mengelola kata sandi pengguna di AWS](#).

Anda dapat mencegah pengguna IAM mengakses AWS Management Console dengan menghapus kata sandi mereka. Hal ini mencegah mereka masuk ke AWS Management Console menggunakan kredensi login mereka. Hal ini tidak mengubah izin mereka atau mencegah mereka mengakses konsol menggunakan peran yang diasumsikan. Jika pengguna memiliki access key aktif, pengguna akan terus berfungsi dan mengizinkan akses melalui AWS CLI, Tools for Windows PowerShell, AWS API, atau AWS Console Mobile Application.

AWS Sumber daya Amazon EC2, wadah Amazon S3, dan seterusnya

Sekalipun pengguna IAM Anda memiliki kata sandi, mereka masih memerlukan izin untuk mengakses AWS sumber daya IAM Anda. Saat Anda membuat pengguna IAM, pengguna tersebut tidak memiliki izin secara default. Untuk memberi pengguna IAM izin yang diperlukan, Anda melampirkan kebijakan pada mereka. Jika Anda memiliki banyak pengguna IAM yang melakukan tugas yang sama dengan sumber daya yang sama, Anda dapat menugaskan pengguna IAM tersebut ke grup. Kemudian tetapkan izin untuk grup tersebut. Untuk informasi tentang membuat pengguna dan grup IAM, lihat [Identitas IAM \(pengguna, pengguna, dan peran\)](#). Untuk informasi tentang menggunakan kebijakan untuk mengatur izin, lihat [Manajemen akses untuk sumber daya AWS](#).

Forum Diskusi AWS

Siapa pun dapat membaca tulisan di [Forum Diskusi AWS](#). Pengguna yang ingin menulis pertanyaan atau komentar ke Forum Diskusi AWS dapat melakukannya dengan menggunakan nama pengguna mereka. Saat pertama kali pengguna memposting ke Forum Diskusi AWS, pengguna diminta memasukkan nama panggilan dan alamat email. Hanya pengguna yang dapat menggunakan nama panggilan tersebut dalam Forum Diskusi AWS.

Informasi Akun AWS penagihan dan penggunaan Anda

Anda dapat memberi pengguna akses ke informasi Akun AWS penagihan dan penggunaan. 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.

Kredensial Akun AWS keamanan Anda

Pengguna tidak dapat mengakses kredensial Akun AWS keamanan Anda.

Note

Kebijakan IAM mengontrol akses terlepas dari antarmuka. Misalnya, Anda dapat memberikan pengguna dengan kata sandi untuk mengakses AWS Management Console. Kebijakan untuk pengguna tersebut (atau grup apa pun yang dimiliki pengguna) akan mengontrol apa yang dapat dilakukan pengguna di AWS Management Console. Atau, Anda dapat memberikan access key AWS pengguna untuk melakukan panggilan API ke AWS. Kebijakan ini akan mengontrol tindakan mana yang dapat pengguna hubungi melalui perpustakaan atau klien yang menggunakan access key tersebut untuk autentikasi.

Cara pengguna IAM M M M IAM IAM IAM IAM IAM AWS

Untuk masuk ke IAM IAM IAM IAM IAM IAM atau alias akun Anda sebagai tambahan ke nama pengguna IAM dan kata sandi Anda. AWS Management Console 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, halaman IAM IAM IAM IAM MAWS masuk menggunakan cookie browser untuk mengingat nama pengguna IAM M M IAM M IAM M IAM IAM IAM IAM M dan informasi akun. Saat berikutnya pengguna IAM masuk ke halaman mana pun di AWS Management Console, konsol menggunakan cookie untuk mengarahkan pengguna IAM IAM masuk ke halaman akses akun.

Anda hanya memiliki akses ke AWS sumber daya yang ditentukan administrator Anda IAM yang terlampir pada identitas pengguna IAM Anda M. Untuk bekerja di konsol, Anda harus memiliki izin untuk melakukan tindakan yang dilaksanakan konsol, seperti mendaftar dan membuat AWS sumber daya. Untuk informasi lebih lanjut, lihat [Manajemen akses untuk sumber daya AWS](#) 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 AWS Management Console untuk akun Anda tanpa mengharuskan mereka memiliki identitas pengguna IAM. SSO juga menghilangkan keharusan pengguna untuk masuk ke situs organisasi Anda dan ke AWS secara terpisah. Untuk informasi selengkapnya, lihat [Mengaktifkan akses broker identitas khusus ke konsol AWS](#).

Mencatat detail masuk di CloudTrail

Jika Anda mengaktifkan CloudTrail untuk mencatat kejadian masuk ke catatan Anda, Anda harus mengetahui cara CloudTrail memilih tempat untuk mencatat kejadian.

- 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 IAM di log wilayah pengguna IAM:

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 kejadian CloudTrail catatan regional.

Untuk informasi lebih lanjut tentang CloudTrail IAM IAM IAM IAM IAM IAM IAM [IAM IAM IAM IAM IAM IAM IAM CloudTrail](#)

Jika pengguna memerlukan akses terprogram untuk bekerja dengan akun Anda, Anda dapat membuat key pair 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.

Menggunakan perangkat MFA dengan halaman masuk IAM Anda

Pengguna yang dikonfigurasi dengan perangkat [otentikasi multi-faktor \(MFA\) harus](#) menggunakan perangkat MFA mereka untuk masuk ke perangkat MFA. AWS Management Console Setelah pengguna memasukkan kredensialnya masuk, AWS memeriksa akun pengguna untuk melihat apakah MFA diperlukan untuk pengguna tersebut. Topik berikut memberikan informasi tentang bagaimana pengguna menyelesaikan masuk saat MFA diperlukan.

Topik

- [Masuk dengan beberapa perangkat MFA diaktifkan](#)
- [Masuk dengan kunci keamanan FIDO](#)
- [Masuk dengan perangkat MFA virtual](#)
- [Masuk dengan token TOTP perangkat keras](#)

Masuk dengan beberapa perangkat MFA diaktifkan

Jika pengguna masuk AWS Management Console 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.

Masuk dengan 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 keamanan FIDO](#)

Masuk dengan 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 AWS Management Console. Jika kode salah, pengguna dapat mencoba kembali dengan kode lain.

Perangkat MFA virtual bisa jadi tidak sinkron. Jika pengguna tidak dapat masuk ke AWS Management Console 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 AndaAkun AWS, lihat [Mengsinkronisasi ulang perangkat MFA virtual dan perangkat keras](#).

Masuk dengan 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 AWS Management Console. Jika kode salah, pengguna dapat mencoba kembali dengan kode lain.

Token TOTP perangkat keras bisa tidak sinkron. Jika pengguna tidak dapat masuk AWS Management Console 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 AndaAkun AWS, lihat [Mengsinkronisasi ulang perangkat MFA virtual dan perangkat keras](#).

Mengelola pengguna IAM

Note

Sebagai [praktik terbaik](#), kami menyarankan Anda meminta pengguna manusia untuk menggunakan federasi dengan penyedia identitas untuk mengakses AWS menggunakan kredensial sementara. Jika Anda mengikuti praktik terbaik, Anda tidak mengelola pengguna dan grup IAM. Sebagai gantinya, 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 kredensial 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 IAM Identity Center](#) di Panduan AWS IAM Identity Center Pengguna untuk informasi tentang membuat pengguna dan grup di IAM Identity Center.

Amazon Web Services menawarkan beberapa alat untuk mengelola pengguna IAM di akun IAM AndaAkun AWS. Anda dapat membuat daftar pengguna IAM di akun atau grup pengguna Anda, atau membuat daftar semua grup pengguna tempat pengguna berada. 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](#), [Menyunting Kebijakan IAM](#), dan [Menghapus kebijakan IAM](#). Sebagai praktik terbaik, gunakan kebijakan terkelola alih-alih kebijakan inline. AWS Kebijakan terlampola memberikan izin bagi banyak kasus penggunaan umum. Perlu diingat bahwa kebijakan AWS terkelola mungkin tidak memberikan izin hak istimewa untuk kasus penggunaan spesifik Anda karena tersedia untuk digunakan oleh semua AWS pelanggan. Sebagai hasilnya, kami menyarankan Anda mengurangi izin lebih lanjut dengan menentukan [kebijakan yang dikelola pelanggan](#) yang spesifik untuk kasus penggunaan Anda. Untuk informasi selengkapnya, lihat [AWS Kebijakan yang dikelola](#). Untuk informasi selengkapnya tentang kebijakan AWS terkelola yang dirancang untuk fungsi pekerjaan tertentu, lihat [AWS kebijakan terkelola untuk fungsi pekerjaan](#).

Untuk mempelajari cara memvalidasi kebijakan IAM, lihat [Memvalidasi kebijakan IAM](#).

Tip

[IAM Access Analyzer](#) dapat menganalisis layanan dan tindakan yang digunakan peran IAM Anda, dan kemudian membuat kebijakan berbutir halus yang dapat Anda gunakan. Setelah menguji setiap kebijakan yang dihasilkan, Anda dapat menerapkan kebijakan 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 IAM Access Analyzer](#).

Untuk informasi tentang pengelolaan kata sandi pengguna IAM, lihat [Mengelola kata sandi untuk pengguna IAM](#),

Topik

- [Lihat akses pengguna](#)
- [Mendaftar pengguna IAM](#)
- [Mengubah nama pengguna IAM](#)
- [Menghapus pengguna IAM](#)
- [Menonaktifkan pengguna IAM](#)

Lihat akses pengguna

Sebelum menghapus 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 [Menyaring izin di AWS menggunakan informasi yang terakhir diakses](#).

Mendaftar pengguna IAM

Anda dapat membuat daftar pengguna IAM di grup pengguna IAM tertentu, dan membuat daftar semua grup pengguna tempat pengguna berada. Akun AWS Untuk informasi tentang izin yang Anda butuhkan untuk membuat daftar pengguna, lihat [Izin diperlukan untuk mengakses sumber daya IAM](#).

Untuk mencantumkan semua pengguna di akun

- [AWS Management Console](#): Di panel navigasi, pilih Pengguna. Konsol menampilkan pengguna di akun Anda.
- AWS CLI: [aws iam list-users](#)
- AWSAPI: [ListUsers](#)

Untuk mencantumkan pengguna dalam grup pengguna tertentu

- [AWS Management Console](#): Di panel navigasi, pilih Grup pengguna, pilih nama grup pengguna, lalu pilih tab Pengguna.
- AWS CLI: [aws iam get-group](#)
- AWSAPI: [GetGroup](#)

Untuk mencantumkan semua grup pengguna tempat pengguna berada

- [AWS Management Console](#): Di panel navigasi, pilih Pengguna, pilih nama pengguna, lalu pilih tab Grup.
- AWS CLI: [AWS list-groups-for-user](#)
- AWSAPI: [ListGroupsForUser](#)

Mengubah nama pengguna IAM

Untuk mengubah nama atau alur pengguna, Anda harus menggunakan alat untuk Windows PowerShell, atau AWS API. AWS CLI 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 pengguna yang sama dengan nama baru.
- ID unik untuk pengguna tetap sama. Untuk informasi selengkapnya tentang ID unik, lihat [Pengidentifikasi 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](#)

- AWSAPI: [UpdateUser](#)

Menghapus pengguna IAM

Anda dapat menghapus pengguna IAM dari penggunaAkun AWS jika pengguna berhenti bekerja di perusahaan Anda. Jika pengguna pergi sementara, Anda dapat menonaktifkan akses pengguna alih-alih menghapusnya dari akun seperti yang dijelaskan di [Menonaktifkan pengguna IAM](#).

Topik

- [Menghapus pengguna IAM \(konsol\)](#)
- [Menghapus pengguna IAM \(AWS CLI\)](#)

Menghapus pengguna IAM (konsol)

Saat Anda menggunakan pengguna IAM, IAM secara otomatis menghapus informasi berikut untuk Anda: AWS Management Console

- Pengguna
- Setiap keanggotaan grup pengguna —yaitu, pengguna dihapus dari grup pengguna IAM yang menjadi anggota pengguna IAM
- Semua kata sandi yang terkait dengan pengguna
- Semua kunci akses milik pengguna
- Semua kebijakan inline yang disertakan di pengguna (kebijakan yang diterapkan ke pengguna melalui izin grup pengguna tidak terpengaruh)

Note

IAM menghapus kebijakan terlampir ke pengguna saat Anda menghapus pengguna, namun tidak menghapus kebijakan terlampir.

- Semua perangkat MFA yang terkait

Untuk menghapus pengguna IAM (konsol)

1. Masuk ke AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.

2. Di panel navigasi, pilih Pengguna, lalu pilih kotak centang di samping nama pengguna yang ingin Anda hapus.
3. Pilih Hapus di bagian atas halaman.
4. Dalam kotak dialog konfirmasi, masukkan nama pengguna dalam bidang input teks untuk mengonfirmasi penghapusan pengguna. Pilih Hapus.

Menghapus pengguna IAM (AWS CLI)

Tidak seperti AWS Management Console, saat Anda menghapus pengguna dengan AWS CLI, Anda harus menghapus item yang melekat ke pengguna secara manual. Prosedur ini menggambarkan proses.

Untuk menghapus pengguna dari akun Anda (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 pengguna mana pun.

[aws iam list-groups-for-user](#) (untuk membuat daftar grup pengguna tempat pengguna berada) 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 menjauh dari perusahaan Anda. Anda dapat meninggalkan kredensi pengguna IAM mereka di tempat dan masih memblokir AWS akses mereka.

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 aksesnya.

Kebijakan berikut tidak menyertakan batas waktu. Anda harus menghapus kebijakan untuk memulihkan akses pengguna.

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

```
{
  "Effect": "Deny",
  "Action": "*",
  "Resource": "*"
}
```

Kebijakan berikut mencakup ketentuan yang memulai kebijakan pada 24 Desember 2024 pukul 23.59 WIB dan mengakhirinya pada 28 Februari 2025 pukul 23.59 WIB.

```
{
  "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"}
      }
    }
  ]
}
```

Mengubah izin untuk pengguna IAM

Anda dapat mengubah izin untuk pengguna IAM di akun AndaAkun AWS dengan mengubah keanggotaan grup, dengan menyalin izin dari pengguna yang sudah ada,, dengan mengatur [batas izin](#). Batas izin mengontrol izin maksimum yang dapat dimiliki pengguna. Batas izin adalah fitur lanjutan AWS.

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 \(konsol\)](#)
- [Menghapus kebijakan izin dari pengguna \(konsol\)](#)
- [Menghapus batas izin dari pengguna \(konsole\)](#)
- [Menambahkan dan menghapus izin pengguna \(AWS CLI atau API AWS\)](#)

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 [Menyaring izin di AWS menggunakan informasi yang terakhir diakses](#).

Menghasilkan kebijakan berdasarkan aktivitas akses pengguna

Terkadang Anda dapat memberikan izin ke entitas IAM (pengguna atau peran) melampaui izin yang mereka butuhkan. Untuk membantu menyempurnakan izin yang diberikan, 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 jangka waktu yang ditentukan. Anda dapat menggunakan templat untuk membuat kebijakan terkelola dengan izin mendetail dan kemudian melampirkannya ke entitas IAM. Dengan begitu, Anda hanya memberikan izin yang pengguna atau peran perlu berinteraksi dengan sumber daya AWS untuk kasus penggunaan spesifik Anda. Untuk pelajari selengkapnya, lihat [Hasilkan kebijakan berdasarkan aktivitas akses](#).

Menambahkan izin ke pengguna (konsol)

IAM menawarkan tiga cara untuk menambahkan kebijakan izin kepada pengguna:

- Tambahkan pengguna ke grup – Buat pengguna menjadi anggota kelompok. Kebijakan dari grup melekat ke pengguna.
- Salin izin dari pengguna yang sudah ada – Salin semua keanggotaan grup, kebijakan terkelola terlampir, kebijakan inline, dan batasan izin yang ada dari pengguna sumber.
- Lekatkan kebijakan secara langsung ke pengguna – Lekatkan kebijakan terkelola langsung ke pengguna. Untuk manajemen izin yang lebih mudah, lampirkan kebijakan Anda pada grup dan kemudian anggota pengguna dari grup yang sesuai.


⚠ Important

Jika pengguna memiliki batas izin, maka Anda tidak dapat menambahkan izin lebih banyak kepada pengguna daripada yang diizinkan oleh batas izin.

Menambahkan izin dengan menambahkan pengguna ke grup


Menambahkan pengguna ke grup akan segera memengaruhi pengguna.

Untuk menambahkan izin ke pengguna dengan menambahkan pengguna ke grup

1. Masuk ke AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Di panel navigasi, pilih Pengguna.
3. Tinjau keanggotaan kelompok saat ini untuk pengguna di kolom Grup pada konsol. Jika perlu, tambahkan kolom ke tabel pengguna dengan menyelesaikan langkah-langkah berikut:
 1. Di atas tabel di ujung kanan, pilih simbol pengaturan ().
 2. Di kotak dialog Kelola Kolom, pilih kolom Grup. Atau, Anda juga dapat mengosongkan kotak centang untuk setiap judul kolom yang tidak ingin Anda tampilkan di tabel pengguna.
 3. Pilih Tutup untuk kembali ke daftar pengguna.

Kolom Grup akan memberi tahu Anda grup mana yang dimiliki pengguna. Kolom tersebut mencakup nama grup untuk maksimal dua grup. Jika pengguna adalah anggota dari tiga grup atau lebih, dua grup pertama ditampilkan (diurutkan secara alfabetik), dan jumlah keanggotaan grup tambahan disertakan. Misalnya, jika pengguna termasuk dalam Grup A, Grup B, Grup C, dan Grup D, bidang tersebut berisi nilai Grup A, Grup B + 2 lagi. Untuk melihat jumlah total grup tempat pengguna berada, Anda dapat menambahkan kolom Jumlah grup ke tabel pengguna.

4. Pilih nama pengguna yang izinnya ingin Anda modifikasi.
5. Pilih tab Izin, lalu pilih Tambahkan izin. Pilih Tambahkan pengguna ke grup.
6. Pilih kotak centang untuk setiap grup yang ingin Anda gabungkan pengguna. Daftar ini menunjukkan nama masing-masing grup dan kebijakan yang diterima pengguna jika telah menjadi anggota grup tersebut.

7. (Opsional) Sebagai tambahan untuk memilih dari grup yang ada, Anda dapat memilih Buat grup untuk menentukan grup baru:
 - a. Di tab baru, untuk Nama grup, ketikkan nama untuk grup baru Anda.
-  **Note**

Jumlah dan ukuran sumber daya IAM di 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 lebih kotak centang untuk kebijakan terkelola yang ingin Anda lampirkan 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 [Membuat kebijakan IAM](#).
 - c. Pilih Buat grup pengguna.
 - d. Kembali ke tab asal, muat ulang daftar grup Anda. Lalu pilih kotak centang untuk grup baru Anda.
 8. Pilih Selanjutnya untuk melihat daftar keanggotaan grup yang akan ditambahkan ke pengguna. Lalu, pilih Tambahkan izin.

Menambahkan izin dengan menyalin dari pengguna lain

Menyalin izin akan segera memengaruhi pengguna.

Untuk menambahkan izin ke pengguna dengan menyalin izin dari pengguna lain

1. Masuk ke AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Pilih Pengguna di panel navigasi, pilih nama pengguna yang izinnya ingin Anda ubah, lalu pilih tab Izin.
3. Pilih Tambahkan izin, lalu pilih Salin izin dari pengguna yang ada. Daftar ini menampilkan pengguna yang tersedia di sepanjang keanggotaan kelompok mereka dan kebijakan yang

melekat. Jika daftar lengkap grup atau kebijakan tidak pas di satu baris, Anda dapat memilih tautan untuk dan *n* banyak lagi. Tindakan tersebut membuka tab browser baru dan melihat daftar kebijakan lengkap (Izin) dan grup (tab Grup).

4. Pilih tombol radio di samping pengguna yang izinnya ingin Anda salin.
5. Pilih Selanjutnya untuk melihat daftar perubahan yang harus dilakukan kepada pengguna. Lalu, pilih Tambahkan izin.

Menambahkan izin dengan melampirkan kebijakan secara langsung ke pengguna

Melampirkan kebijakan akan segera memengaruhi pengguna.

Untuk menambahkan izin kepada pengguna dengan melampirkan secara langsung kebijakan yang dikelola

1. Masuk ke AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Pilih Pengguna di panel navigasi, pilih nama pengguna yang izinnya ingin Anda ubah, lalu pilih tab Izin.
3. Pilih Tambahkan izin, lalu pilih Lampirkan kebijakan secara langsung.
4. Pilih satu kotak centang atau lebih untuk kebijakan terkelola yang ingin Anda lampirkan ke pengguna. Anda juga dapat membuat kebijakan pengelolaan baru dengan memilih Buat kebijakan. Jika Anda melakukannya, kembali ke tab atau jendela browser ini jika kebijakan baru sudah selesai. Pilih Segarkan; dan kemudian pilih kotak centang untuk kebijakan baru untuk dilampirkan ke pengguna Anda. Untuk informasi selengkapnya, lihat [Membuat kebijakan IAM](#) .
5. Pilih Selanjutnya untuk melihat daftar kebijakan yang akan dilampirkan pada pengguna. Lalu, pilih Tambahkan izin.

Mengatur batas izin untuk pengguna

Pengaturan batas izin akan segera memengaruhi pengguna.

Untuk menetapkan batas izin bagi pengguna

1. Masuk ke AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Di panel navigasi, pilih Pengguna.

3. Pilih nama pengguna yang memiliki batas izin yang ingin Anda ubah.
4. Pilih tab Izin. Jika perlu, buka bagian Batas izin lalu pilih Tetapkan batas izin.
5. Pilih kebijakan yang ingin Anda gunakan untuk batasan izin.
6. Pilih Tetapkan batas.

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 akan segera memengaruhi pengguna.

Untuk mengedit kebijakan terkelola yang dilampirkan pada pengguna

1. Masuk ke AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Di panel navigasi, pilih Pengguna.
3. Pilih nama pengguna yang memiliki kebijakan izin yang ingin Anda ubah.
4. Pilih tab Izin. Jika perlu, buka bagian Kebijakan izin.
5. Pilih nama kebijakan yang ingin Anda edit untuk melihat perincian tentang kebijakan tersebut. Pilih tab Penggunaan kebijakan untuk melihat entitas lainnya yang mungkin terpengaruh jika Anda mengedit kebijakan.
6. Pilih Tab Izin dan meninjau izin yang diberikan oleh kebijakan. Lalu, pilih Edit kebijakan.
7. Edit kebijakan dan selesaikan rekomendasi [validasi kebijakan](#) apa pun. Untuk informasi selengkapnya, lihat [Menyunting Kebijakan IAM](#).
8. Pilih Tinjau kebijakan, tinjau ringkasan kebijakan, lalu pilih Simpan perubahan.

Mengubah batas izin untuk pengguna

Mengubah batas izin akan segera memengaruhi pengguna.

Untuk mengubah kebijakan yang digunakan untuk mengatur batas izin bagi pengguna

1. Masuk ke AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Di panel navigasi, pilih Pengguna.
3. Pilih nama pengguna yang memiliki 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 Tetapkan batas.

Menghapus kebijakan izin dari pengguna (konsol)

Menghapus kebijakan akan segera memengaruhi pengguna.

Untuk menghapus izin bagi pengguna IAM

1. Masuk ke AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Di panel navigasi, pilih Pengguna.
3. Pilih nama pengguna yang memiliki batas izin yang ingin Anda hapus.
4. Pilih tab Izin.
5. Jika Anda ingin menghapus izin dengan menghapus kebijakan yang sudah ada, lihat Jenis untuk memahami bagaimana pengguna memperoleh kebijakan tersebut sebelum memilih Hapus untuk menghapus kebijakan:
 - Jika kebijakan berlaku karena keanggotaan grup, maka pilih 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 adalah kebijakan terkelola yang dilampirkan langsung ke pengguna, maka pilih Hapus akan melepaskan kebijakan dari pengguna. Hal ini tidak memengaruhi kebijakan itu sendiri atau entitas lainnya yang mungkin menyertai kebijakan tersebut.

- Jika kebijakan tersebut adalah kebijakan tertanam inline, maka memilih X akan menghapus kebijakan dari IAM. Kebijakan inline yang dilampirkan langsung ke pengguna hanya ada pada pengguna tersebut.

Menghapus batas izin dari pengguna (konsole)

Menghapus batas izin akan segera memengaruhi pengguna.

Untuk menghapus batas izin dari pengguna

1. Masuk ke AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Di panel navigasi, pilih Pengguna.
3. Pilih nama pengguna yang memiliki batas izin yang ingin Anda hapus.
4. Pilih tab Izin. Jika perlu, buka bagian Batas izin lalu pilih Hapus batas.
5. Pilih Hapus batas untuk mengonfirmasi bahwa Anda ingin menghapus batas izin.

Menambahkan dan menghapus izin pengguna (AWS CLI atau API AWS)

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:

- [Menambahkan dan menghapus pengguna di grup pengguna IAM](#)
- [Menambahkan dan menghapus izin identitas IAM](#)

Mengelola kata sandi pengguna di AWS

Anda dapat mengelola kata sandi untuk pengguna IAM di akun Anda. Pengguna IAM membutuhkan kata sandi untuk mengakses AWS Management Console. Pengguna tidak memerlukan kata sandi untuk mengakses AWS sumber daya pemrograman dengan menggunakan AWS CLI, Alat untuk Windows PowerShell, yang AWSSDK atau API. Untuk lingkungan tersebut, Anda memiliki opsi untuk menetapkan pengguna IAM [kunci akses](#). Namun, ada alternatif lain yang lebih aman untuk mengakses kunci yang kami sarankan Anda pertimbangkan terlebih dahulu. Untuk informasi selengkapnya, lihat [Kredensial keamanan AWS](#).

Konten

- [Mengatur kebijakan kata sandi akun untuk pengguna IAM](#)
- [Mengelola kata sandi untuk pengguna IAM](#)
- [Mengizinkan pengguna IAM untuk mengubah kata sandi mereka sendiri](#)
- [Bagaimana pengguna IAM mengubah kata sandi mereka sendiri](#)

Mengatur 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](#)
- [Mengatur kebijakan kata sandi \(konsol\)](#)
- [Mengatur kebijakan kata sandi \(AWS CLI\)](#)
- [Mengatur kebijakan kata sandi \(API AWS\)](#)

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 AWS Management Console 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 [Menggunakan otentikasi multi-faktor \(MFA\) di AWS](#).

Izin yang diperlukan untuk menetapkan kebijakan kata sandi

Anda harus mengonfigurasi 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”](#).

```
{
  "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 [Mengizinkan 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 AWS Management Console 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 AWS Management Console 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. AWS Management Console
- 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, kosongkan kotak centang ini. Untuk informasi selengkapnya tentang penggunaan kebijakan untuk membatasi siapa yang dapat mengelola kata sandi, lihat [Mengizinkan 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.

Mengatur kebijakan kata sandi (konsol)

Anda dapat menggunakan AWS Management Console untuk membuat, mengubah, atau menghapus kebijakan kata sandi khusus.

Untuk membuat kebijakan kata sandi khusus (konsol)

1. Masuk ke AWS Management Console 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 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.

Untuk mengubah kebijakan kata sandi khusus (konsol)

1. Masuk ke AWS Management Console 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 opsi yang ingin Anda terapkan ke kebijakan kata sandi Anda dan pilih Simpan perubahan.
5. Konfirmasikan bahwa Anda ingin menetapkan kebijakan kata sandi khusus dengan memilih Setel kustom.

Untuk menghapus kebijakan kata sandi khusus (konsol)

1. Masuk ke AWS Management Console 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 IAM default untuk menghapus kebijakan kata sandi kustom dan pilih Simpan perubahan.
5. Konfirmasikan bahwa Anda ingin mengatur kebijakan kata sandi default IAM dengan memilih Set default.

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](#)

Mengatur kebijakan kata sandi (API AWS)

Anda dapat menggunakan operasi API AWS untuk mengatur kebijakan kata sandi.

Untuk mengelola kebijakan kata sandi akun kustom dari API AWS

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](#)

Mengelola kata sandi untuk pengguna IAM

Pengguna IAM yang menggunakan AWS Management Console untuk bekerja dengan AWS sumber daya harus memiliki kata sandi agar dapat masuk. Anda dapat membuat, mengubah, atau menghapus kata sandi untuk pengguna IAM di AWS akun Anda.

Setelah Anda menetapkan kata sandi ke pengguna, pengguna dapat masuk ke AWS Management Console 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 AWS Management Console, lihat [Cara masuk ke AWS dalam](#) 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 \(pengguna, pengguna, dan peran\)](#). 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 [Mengizinkan pengguna IAM untuk mengubah kata sandi mereka sendiri](#). Untuk informasi tentang cara pengguna mengakses halaman masuk akun Anda, lihat [Cara masuk keAWS dalam](#) PanduanAWS Sign-In Pengguna.

Topik

- [Membuat, mengubah, atau menghapus kata sandi pengguna IAM \(konsol\)](#)
- [Membuat, mengubah, atau menghapus kata sandi pengguna IAMAWS CLI](#)
- [Membuat, mengubah, atau menghapus kata sandi pengguna IAM \(AWSAPI\) kata sandi pengguna IAM](#)

Membuat, mengubah, atau menghapus kata sandi pengguna IAM (konsol)


Anda dapat menggunakanAWS Management Console untuk mengelola kata sandi untuk pengguna IAM Anda.

Ketika pengguna meninggalkan organisasi Anda atau tidak lagi memerlukan akses AWS, penting untuk menemukan kredensial yang mereka gunakan dan memastikan bahwa mereka tidak lagi beroperasi. Idealnya, Anda menghapus kredensial jika tidak lagi diperlukan. Anda dapat selalu membuat ulang catatan di kemudian hari jika perlu. Setidaknya, Anda harus mengubah kredensial sehingga pengguna sebelumnya tidak lagi memiliki akses.

Untuk menambahkan kata sandi bagi pengguna IAM (konsol)


1. Masuk ke AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Di panel navigasi, pilih Pengguna.
3. Pilih nama pengguna yang kata sandinya ingin Anda buat.
4. Pilih tab Kredensi keamanan, lalu di bagian Login dengan konsol, pilih Aktifkan akses konsol.
5. Di Mengelola akses konsol, untuk Akses konsol pilih Aktifkan jika belum dipilih. Jika akses konsol dinonaktifkan, maka sandi tidak diperlukan.
6. Untuk Atur kata sandi, pilih apakah membuat kata sandi atau membuat kata sandi khusus untuk IAM:

- 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 meminta pengguna membuat kata sandi baru saat masuk, pilih Pengguna harus membuat kata sandi baru saat masuk berikutnya. Lalu, pilih Terapkan.

 Important

Jika Anda memilih Pengguna harus membuat kata sandi baru saat masuk berikutnya, pastikan bahwa pengguna memiliki izin untuk mengubah kata sandinya. Untuk informasi selengkapnya, lihat [Mengizinkan pengguna IAM untuk mengubah kata sandi mereka sendiri](#).

8. Jika Anda memilih opsi untuk membuat kata sandi, pilih Tampilkan di kotak dialog kata sandi. Hal ini memungkinkan Anda melihat kata sandi sehingga Anda dapat membagikannya dengan pengguna.


 Important

Untuk alasan keamanan, Anda tidak dapat mengakses kata sandi setelah menyelesaikan langkah ini, tetapi Anda dapat membuat kata sandi baru kapan saja.

Untuk mengubah kata sandi bagi pengguna IAM (konsol)

1. Masuk ke AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Di panel navigasi, pilih Pengguna.
3. Pilih nama pengguna yang kata sandinya ingin Anda ubah.
4. Pilih tab Kredensi keamanan, lalu di bagian Login dengan konsol, pilih Kelola akses konsol.
5. Di Mengelola akses konsol, untuk Akses konsol pilih Aktifkan jika belum dipilih. Jika akses konsol dinonaktifkan, maka sandi tidak diperlukan.

6. Untuk Atur kata sandi, pilih apakah membuat kata sandi atau membuat kata sandi khusus untuk IAM:
 - 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](#), jika saat ini diatur.

7. Untuk meminta pengguna membuat kata sandi baru saat masuk, pilih Pengguna harus membuat kata sandi baru saat masuk berikutnya. Lalu, pilih Terapkan.

 Important

Jika Anda memilih Pengguna harus membuat kata sandi baru saat masuk berikutnya, pastikan bahwa pengguna memiliki izin untuk mengubah kata sandinya. Untuk informasi selengkapnya, lihat [Mengizinkan pengguna IAM untuk mengubah kata sandi mereka sendiri](#).

8. Jika Anda memilih opsi untuk membuat kata sandi, pilih Tampilkan di kotak dialog kata sandi. Hal ini memungkinkan Anda melihat kata sandi sehingga Anda dapat membagikannya dengan pengguna.

 Important

Untuk alasan keamanan, Anda tidak dapat mengakses kata sandi setelah menyelesaikan langkah ini, tetapi Anda dapat membuat kata sandi baru kapan saja.

Untuk menghapus (menonaktifkan) kata sandi pengguna IAM (konsol) kata sandi pengguna IAM

1. Masuk ke AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Di panel navigasi, pilih Pengguna.
3. Pilih nama pengguna yang kata sandinya ingin Anda hapus.
4. Pilih tab Kredensi keamanan, lalu di bagian Login dengan konsol, pilih Kelola akses konsol.

5. Untuk Akses konsol, pilih Nonaktifkan, lalu pilih Terapkan.

Important

Anda dapat mencegah pengguna IAM dari mengakses AWS Management Console dengan menghapus kata sandinya. Hal ini mencegah mereka dari masuk ke AWS Management Console menggunakan kredensi masuk mereka. Hal ini tidak mengubah izin mereka atau mencegah mereka mengakses konsol menggunakan peran yang diasumsikan. Jika pengguna memiliki tombol akses aktif, pengguna akan terus berfungsi dan memungkinkan akses melalui AWS CLI, Tools for Windows, AWS API PowerShell, atau AWS Console Mobile Application.

Membuat, mengubah, atau menghapus kata sandi pengguna IAM AWS CLI

Anda dapat menggunakan AWS CLI API untuk mengelola kata sandi 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 AWS Management Console. Jika pengguna memiliki tombol akses aktif, pengguna akan terus berfungsi dan memungkinkan akses melalui AWS CLI, Tools for Windows PowerShell, atau panggilan fungsi AWS API. Saat Anda menggunakan AWS CLI, Tools for Windows PowerShell, atau AWS API untuk menghapus pengguna dari Akun AWS, Anda harus terlebih dahulu menghapus kata sandi menggunakan operasi ini. Untuk informasi selengkapnya, lihat [Menghapus pengguna IAM \(AWS CLI\)](#).

Membuat, mengubah, atau menghapus kata sandi pengguna IAM (AWS API) kata sandi pengguna IAM

Anda dapat menggunakan AWS API untuk mengelola kata sandi untuk mengelola kata sandi untuk pengguna IAM Anda.

Untuk membuat kata sandi (API AWS)

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 (API AWS)

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 (API AWS)

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 AWS Management Console. Jika pengguna memiliki tombol akses aktif, pengguna akan terus berfungsi dan memungkinkan akses melalui AWS CLI, Tools for Windows PowerShell, atau panggilan fungsi AWS API. Saat Anda menggunakan AWS CLI, Tools for Windows PowerShell, atau AWS API untuk menghapus pengguna dari Akun AWS, Anda harus terlebih dahulu menghapus kata sandi menggunakan operasi ini. Untuk informasi selengkapnya, lihat [Menghapus pengguna IAM \(AWS CLI\)](#).

Mengizinkan pengguna IAM untuk mengubah kata sandi mereka sendiri

ℹ 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 kredensi sementara.

Anda dapat memberikan izin kepada pengguna IAM untuk mengubah kata sandi mereka sendiri untuk masuk ke AWS Management Console 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 kata 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 pengguna IAM mengubah kata sandi mereka sendiri

1. Masuk ke AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Di panel navigasi, klik 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](#).

Untuk informasi tentang perintah AWS CLI, Alat untuk Windows PowerShell, dan API yang dapat Anda gunakan untuk mengubah kebijakan kata sandi akun (termasuk membiarkan semua pengguna mengubah kata sandi mereka sendiri), lihat [Mengatur kebijakan kata sandi \(AWS CLI\)](#).

Untuk mengizinkan pengguna IAM terpilih untuk mengubah kata sandi mereka sendiri

1. Masuk ke AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Di panel navigasi, klik 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. Untuk detailnya, lihat [Membuat pengguna IAM di Akun AWS](#).
5. (Opsional) Buat grup IAM untuk pengguna yang diizinkan untuk mengubah kata sandi mereka, lalu tambahkan pengguna dari langkah sebelumnya ke grup. Untuk detailnya, lihat [Mengelola grup pengguna IAM](#).
6. Tetapkan kebijakan berikut kepada grup. Untuk informasi selengkapnya, lihat [Mengelola kebijakan IAM](#).

```
{
  "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 untuk 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:

- [Mengizinkan pengguna IAM untuk mengubah kata sandi mereka sendiri](#)
- [Mengelola kata sandi pengguna di AWS](#)
- [Mengatur kebijakan kata sandi akun untuk pengguna IAM](#)
- [Mengelola 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 AWS Management Console untuk melakukan ini. Anda juga dapat menggunakan AWS CLI atau API AWS.

Topik

- [Izin diperlukan](#)
- [Cara pengguna IAM mengubah kata sandi mereka sendiri \(konsol\)](#)
- [Bagaimana pengguna IAM mengubah kata sandi \(AWS CLI atau AWS API\) mereka sendiri](#)

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 AWS Management Console untuk mengubah kata sandi mereka sendiri.

Untuk mengubah kata sandi pengguna IAM Anda sendiri (konsol)

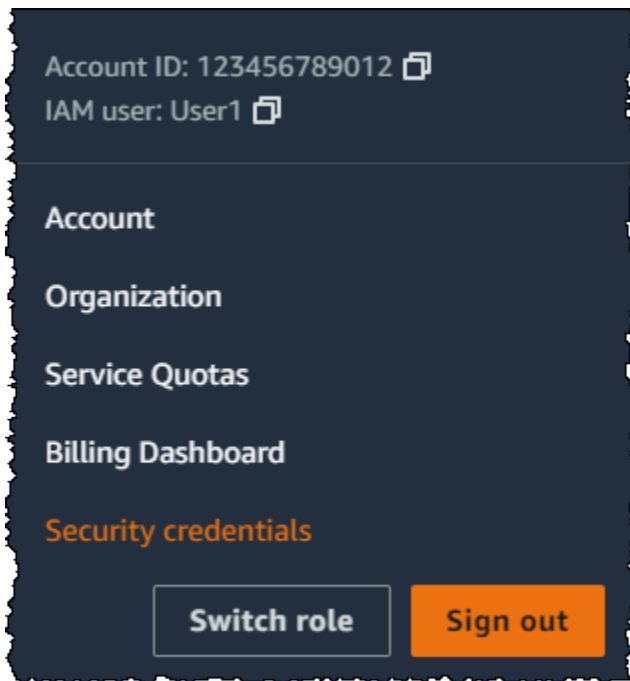
1. Gunakan ID AWS akun atau alias akun, nama pengguna IAM, dan kata sandi Anda untuk masuk ke konsol [IAM](#).

Note

Untuk kenyamanan Anda, halaman masuk AWS menggunakan cookie browser untuk mengingat nama pengguna dan informasi akun pengguna IAM. 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 [Mengatur kebijakan kata sandi akun untuk pengguna IAM](#).

Bagaimana pengguna IAM mengubah kata sandi (AWS CLI atau AWS API) mereka sendiri

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](#)

Mengelola kunci akses untuk pengguna IAM

[Follow us on Twitter](#)

⚠ Important

Sebagai [praktik terbaik](#), gunakan kredensial keamanan sementara (seperti peran IAM) alih-alih membuat kredensial jangka panjang seperti kunci akses. Sebelum membuat kunci akses, tinjau [alternatif untuk kunci akses jangka panjang](#).

Kunci akses adalah kredensial 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 [Menandatangani permintaan AWS API](#).

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. Secret access key hanya tersedia saat Anda membuatnya. Jika Anda kehilangan secret access key, Anda harus menghapus access key dan membuat access key baru. Untuk lebih detailnya, lihat [Mengatur ulang kata sandi yang hilang atau terlupakan atau access key untuk AWS](#).

Anda dapat memiliki maksimal dua kunci akses per pengguna.

⚠ Important

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.


Topik berikut merinci tugas manajemen yang terkait dengan kunci akses.

Topik

- [Izin diperlukan untuk mengelola kunci akses](#)
- [Mengelola access key \(konsol\)](#)
- [Mengelola access key \(AWS CLI\)](#)

- [Mengelola kunci akses \(AWS API\)](#)
- [Memperbarui kunci akses](#)
- [Mengamankan kunci akses](#)
- [Mengaudit access key](#)

Izin diperlukan untuk mengelola kunci akses

 Note

`iam:TagUser` adalah izin opsional untuk menambahkan dan mengedit deskripsi untuk kunci akses. Lihat informasi yang lebih lengkap di [Menandai pengguna IAM](#)

Untuk membuat kunci akses untuk pengguna IAM Anda sendiri, Anda harus memiliki izin dari kebijakan berikut:

```
{
  "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:

```
{
  "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}"
}
```

Mengelola access key (konsol)

Anda dapat menggunakan AWS Management Console untuk mengelola kunci akses pengguna IAM.

Untuk membuat, memodifikasi, atau menghapus kunci akses Anda sendiri (konsol)

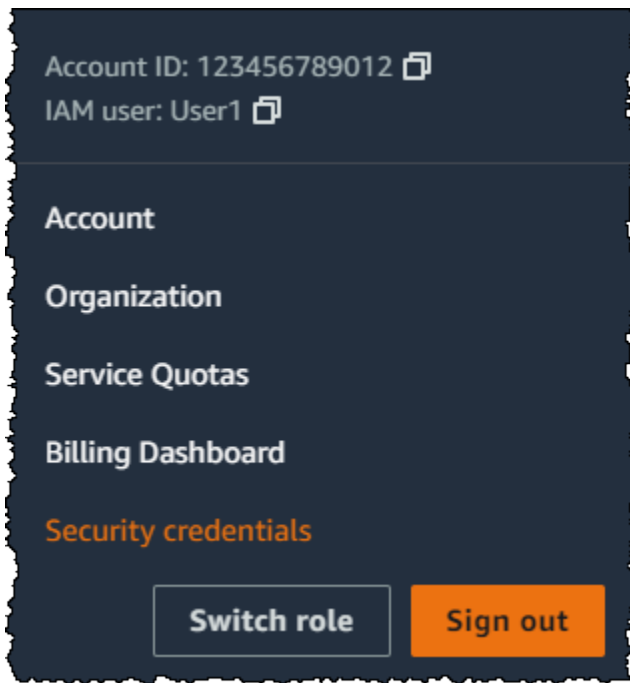
1. Gunakan ID AWS akun atau alias akun, nama pengguna IAM, dan kata sandi Anda untuk masuk ke konsol [IAM](#).

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 menyetel 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.



Lakukan salah satu hal berikut:

Untuk membuat kunci akses

1. 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.
2. 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.
3. (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.
4. 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.

Untuk menonaktifkan kunci akses

- 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.

Untuk mengaktifkan kunci akses

- Di bagian Kunci akses, temukan kunci untuk mengaktifkan, lalu pilih Tindakan, lalu pilih Aktifkan.

Untuk menghapus kunci akses saat Anda tidak lagi membutuhkannya

- 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.


Untuk membuat, memodifikasi, atau menghapus kunci akses pengguna IAM lain (konsol)

1. Masuk ke AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Di panel navigasi, pilih Pengguna.
3. Pilih nama pengguna yang access key-nya ingin Anda kelola, lalu pilih tab Kredensial keamanan.
4. Di bagian Tombol akses, lakukan salah satu hal berikut:
 - Untuk membuat access key, pilih Buat access key. Jika tombol dinonaktifkan, maka Anda harus menghapus salah satu kunci yang ada sebelum Anda dapat membuat yang baru. Pada halaman Praktik & Alternatif Terbaik Kunci Akses, tinjau praktik dan alternatif terbaik. Pilih kasus penggunaan Anda untuk mempelajari tentang opsi tambahan yang dapat membantu Anda menghindari membuat kunci akses jangka panjang. Jika Anda menentukan bahwa kasus penggunaan Anda masih memerlukan kunci akses, pilih Lainnya dan kemudian pilih Berikutnya. Pada halaman Retrieve access key, pilih Tampilkan untuk mengungkapkan nilai kunci akses rahasia pengguna Anda. Untuk menyimpan ID kunci akses dan kunci akses rahasia ke .csv file ke lokasi aman di komputer Anda, pilih tombol Unduh file.csv. Saat Anda membuat kunci akses untuk pengguna Anda, key pair tersebut aktif secara default, dan pengguna Anda dapat langsung menggunakan pasangan tersebut.

- Untuk menonaktifkan kunci akses aktif, pilih Tindakan, lalu pilih Nonaktifkan.
- Untuk mengaktifkan kunci akses tidak aktif, pilih Tindakan, lalu pilih Aktifkan.
- Untuk menghapus kunci akses Anda, pilih Tindakan, lalu pilih Hapus. Ikuti instruksi dalam dialog untuk pertama Nonaktifkan dan kemudian konfirmasi penghapusan. AWS merekomendasikan bahwa sebelum Anda melakukan ini, Anda terlebih dahulu menonaktifkan kunci dan menguji bahwa itu tidak lagi digunakan. Saat Anda menggunakan AWS Management Console, Anda harus menonaktifkan kunci Anda sebelum menghapusnya.


Untuk mencantumkan kunci akses untuk pengguna IAM (konsol)

1. Masuk ke AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Di panel navigasi, pilih Pengguna.
3. Pilih nama pengguna yang dimaksud, lalu pilih tab Kredensial keamanan. Di bagian tombol Akses, Anda akan melihat tombol akses pengguna dan status setiap tombol ditampilkan.

 Note


Hanya access key ID pengguna yang terlihat. Secret access key hanya dapat diambil kembali saat kunci dibuat.

Untuk mencantumkan ID kunci akses untuk beberapa pengguna IAM (konsol)

1. Masuk ke AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Di panel navigasi, pilih Pengguna.
3. Jika perlu, tambahkan kolom access key ID pada tabel pengguna dengan menyelesaikan langkah-langkah berikut:
 - a. Di atas tabel di ujung kanan, pilih ikon pengaturan ().
 - b. Di Kelola kolom, pilih access key ID.
 - c. Pilih Tutup untuk kembali ke daftar pengguna.


4. Kolom access key ID menampilkan setiap access key ID, diikuti oleh statusnya; misalnya, 23478207027842073230762374023 (Aktif) atau 22093740239670237024843420327 (Tidak aktif).

Anda dapat menggunakan informasi ini untuk melihat dan menyalin access key bagi pengguna dengan satu atau dua access key. Kolom menampilkan Tidak ada untuk pengguna yang tidak memiliki access key.

 Note

Hanya access key ID dan status pengguna yang terlihat. Kunci akses rahasia hanya dapat diambil kembali saat kunci dibuat.

Untuk menemukan pengguna IAM mana yang memiliki kunci akses khusus (konsol)

1. Masuk ke AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Di panel navigasi, pilih Pengguna.
3. Di kotak pencarian, ketik atau tempelkan access key ID pengguna yang ingin Anda temukan.
4. Jika perlu, tambahkan kolom access key ID pada tabel pengguna dengan menyelesaikan langkah-langkah berikut:
 - a. Di atas tabel di ujung kanan, pilih ikon pengaturan ).
 - b. Di Kelola kolom, pilih access key ID.
 - c. Pilih Tutup untuk kembali ke daftar pengguna dan mengonfirmasi bahwa pengguna yang difilter memiliki access key tertentu.

Mengelola access key (AWS CLI)

Untuk mengelola kunci akses pengguna IAM dari AWS CLI, jalankan perintah berikut.

- Untuk membuat access key: [aws iam create-access-key](#)
- Untuk menonaktifkan atau mengaktifkan kunci akses: [aws iam update-access-key](#)
- Untuk mencantumkan access key pengguna: [aws iam list-access-keys](#)

- Untuk menentukan kapan access key digunakan baru-baru ini: [aws iam get-access-key-last-used](#)
- Untuk menghapus access key: [aws iam delete-access-key](#)

Mengelola kunci akses (AWS API)

Untuk mengelola kunci akses pengguna IAM dari AWS API, panggil operasi berikut.

- Untuk membuat access key: [CreateAccessKey](#)
- Untuk menonaktifkan atau mengaktifkan kunci akses: [UpdateAccessKey](#)
- Untuk mencantumkan access key pengguna: [ListAccessKeys](#)
- Untuk menentukan kapan access key digunakan baru-baru ini: [GetAccessKeyLastUsed](#)
- Untuk menghapus access key: [DeleteAccessKey](#)

Memperbarui kunci akses

Sebagai [praktik keamanan terbaik](#), kami menyarankan Anda memperbarui kunci akses pengguna IAM saat diperlukan, seperti ketika seorang karyawan meninggalkan perusahaan Anda. Pengguna IAM dapat memperbarui kunci akses mereka sendiri jika mereka telah diberikan izin yang diperlukan.

Untuk detail tentang pemberian izin kepada pengguna IAM untuk memperbarui kunci akses mereka sendiri, lihat [AWS: 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 [Mengatur kebijakan kata sandi akun untuk pengguna IAM](#).

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 file AWS Management Console.

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 AWS Management Console 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 & alternatif terbaik kunci Akses, pilih Lainnya, lalu pilih Berikutnya.
 - e. (Opsional) Tetapkan nilai tag deskripsi untuk kunci akses untuk menambahkan pasangan nilai kunci tag ke pengguna IAM ini. 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.
 - f. 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.

Saat Anda membuat kunci akses untuk pengguna Anda, key pair tersebut aktif secara default, dan pengguna Anda dapat langsung menggunakan pasangan tersebut. 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 yang terakhir 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 yang digunakan menunjukkan bahwa kunci lama tidak pernah digunakan, kami sarankan Anda tidak segera menghapus kunci akses pertama. Sebagai gantinya, pilih Tindakan dan kemudian pilih Nonaktifkan untuk menonaktifkan kunci akses pertama.
5. Gunakan hanya access key baru untuk mengonfirmasi bahwa aplikasi Anda berfungsi. Aplikasi dan alat apa pun yang masih menggunakan kunci akses asli akan berhenti berfungsi 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 Anda menunggu beberapa waktu untuk memastikan bahwa semua aplikasi dan alat telah diperbarui, Anda dapat menghapus access key pertama:
 - a. Masuk ke AWS Management Console 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 Kunci akses untuk kunci akses yang ingin Anda hapus, pilih Tindakan, lalu pilih Hapus. Ikuti instruksi dalam dialog untuk pertama Nonaktifkan dan kemudian konfirmasi penghapusan.

Untuk menentukan kunci akses mana yang perlu diperbarui atau dihapus (konsol)

1. Masuk ke AWS Management Console 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 file 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. Aplikasi dan alat apa pun yang masih menggunakan kunci akses asli akan berhenti berfungsi pada saat ini karena mereka tidak lagi memiliki akses ke AWS sumber daya. Jika Anda menemukan aplikasi atau alat seperti itu, Anda dapat mengubah 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 access key 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 (API)AWS

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. Alih-alih, ubah status access key pertama ke Inactive dengan operasi ini:

- [UpdateAccessKey](#)

5. Gunakan hanya access key baru untuk mengonfirmasi bahwa aplikasi Anda berfungsi. Aplikasi dan alat apa pun yang masih menggunakan kunci akses asli akan berhenti berfungsi pada saat ini karena mereka tidak lagi memiliki akses ke AWS sumber daya. Jika Anda menemukan aplikasi atau alat seperti itu, Anda dapat mengubah 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 access key pertama dengan memanggil operasi ini:

- [DeleteAccessKey](#)

Mengamankan kunci akses

Siapa pun yang memiliki kunci akses Anda memiliki tingkat akses yang sama ke AWS sumber daya Anda seperti yang Anda lakukan. Akibatnya, AWS berupaya 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) kunci Pengguna root akun AWS 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 administratif harian. Untuk informasi tentang cara membuat pengguna administratif di Pusat Identitas IAM, lihat [Memulai](#) di Panduan Pengguna Pusat Identitas IAM.

Jika Anda sudah memiliki kunci akses pengguna root untuk akun Anda, kami sarankan yang berikut: Temukan tempat di aplikasi Anda di mana 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 [Memperbarui kunci akses](#)

Gunakan kredensial 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.

Kunci akses jangka panjang, seperti yang terkait dengan pengguna IAM dan pengguna root, tetap valid sampai Anda mencabutnya secara manual. Namun, kredensi keamanan sementara yang diperoleh melalui peran IAM dan fitur lain dari 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 pada instans Amazon EC2. Jangan gunakan kunci akses langsung di aplikasi Anda. Jangan memberikan access key ke aplikasi, menyimpannya 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 SDK dan AWS Command Line Interface (AWS CLI) bisa mendapatkan 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 AWS ke akun yang menggunakan peran IAM.](#)
- Anda memiliki aplikasi seluler. Jangan menyimpan 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 ke dalam 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 bergabung ke dalam AWS dan organisasi Anda memiliki toko identitas lokal. Jika pengguna dapat mengautentikasi di dalam organisasi Anda, Anda dapat menulis aplikasi yang dapat mengeluarkan kredensial keamanan sementara untuk akses ke sumber daya. AWS Untuk informasi selengkapnya, lihat [Mengaktifkan akses broker identitas khusus ke konsol AWS.](#)

Note

Apakah Anda menggunakan instans Amazon EC2 dengan aplikasi yang memerlukan akses terprogram ke sumber daya? AWS Jika demikian, gunakan [peran IAM untuk EC2.](#)

Kelola kunci akses pengguna IAM dengan benar

Jika Anda harus membuat kunci akses untuk akses terprogram AWS, buat mereka untuk pengguna IAM, memberikan pengguna hanya izin yang mereka butuhkan.

Perhatikan tindakan pencegahan ini untuk membantu melindungi kunci akses pengguna IAM:

- Jangan menanamkan kunci akses langsung ke kode. [AWS SDK](#) dan [AWS Command Line Tools](#) 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 kredensialnya. AWS SDK dan AWS CLI secara otomatis menggunakan kredensial yang Anda simpan di file kredensial. AWS

Untuk informasi tentang menggunakan file AWS kredensial, lihat dokumentasi untuk SDK Anda. Contohnya termasuk [Set AWS Credentials dan Region](#) dalam Panduan AWS SDK for Java Pengembang dan [Konfigurasi dan file kredensi](#) di Panduan Pengguna AWS Command Line Interface

Untuk menyimpan kredensial untuk AWS SDK for .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 for .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 diekspos. 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 tombol akses bila diperlukan. Jika ada risiko bahwa kunci akses dapat dikompromikan, perbarui kunci akses dan hapus kunci akses sebelumnya. Untuk detailnya, lihat [Memperbarui 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 [GetAccessKeyLastUsedAPI](#) (AWS CLI command: [aws iam get-access-key-last-used](#)).

- Gunakan kredensial sementara dan konfigurasi autentikasi 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 mengautentikasi dengan perangkat AWS MFA. Untuk informasi selengkapnya, lihat [Mengonfigurasi akses API yang dilindungi MFA](#).

Akses aplikasi seluler menggunakan tombol AWS 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 bahwa selain menggunakan kata sandi atau kunci biometrik pada 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 kunci AWS CLI akses yang akan digunakan:

- [Menetapkan AWS kredensial dan Wilayah](#) dalam Panduan Pengembang AWS SDK for Java
- [Menggunakan SDK Store](#) di Panduan AWS SDK for .NET Pengembang
- [Memberikan Kredensial ke SDK di Panduan Pengembang AWS SDK for PHP](#)
- [Konfigurasi](#) dalam dokumentasi Boto 3 (AWS SDK untuk Python)
- [Menggunakan AWS Kredensial](#) di Panduan Pengguna AWS Tools for Windows PowerShell
- [File konfigurasi dan kredensi](#) di AWS Command Line Interface Panduan Pengguna
- [Memberikan akses menggunakan peran IAM dalam Panduan](#) Pengembang AWS SDK for .NET
- [Konfigurasi peran IAM untuk Amazon EC2](#) di AWS SDK for Java 2.x

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.

Operasi AWS CLI dan AWS 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. 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 mengetahui siapa yang meminta kredensial sementara untuk kunci ASIA akses, lihat AWS STS peristiwa di log Anda CloudTrail .

Untuk tujuan keamanan, Anda dapat [meninjau AWS CloudTrail log](#) untuk mengetahui siapa yang melakukan tindakan AWS. Anda dapat menggunakan kunci syarat `sts:SourceIdentity` dalam peran kebijakan kepercayaan untuk mengharuskan pengguna menentukan identitas 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](#).

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.

Mengatur ulang kata sandi yang hilang atau terlupakan atau access key untuk AWS

Important

Mengalami masalah saat 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 kredensial yang Anda atur saat membuat Akun AWS. Jika Anda adalah pengguna IAM, administrator akun Anda dapat memberi Anda kredensial yang dapat Anda gunakan untuk masuk. AWS. Jika Anda perlu meminta dukungan, jangan gunakan tautan umpan balik di halaman ini, karena formulir diterima oleh tim Dokumentasi AWS, bukan AWS Support. Sebagai gantinya, pada halaman [Hubungi Kami](#) pilih Masih tidak dapat masuk ke AWS akun Anda dan kemudian pilih salah satu opsi dukungan yang tersedia.

Di halaman masuk utama, Anda harus memasukkan alamat email Anda untuk masuk sebagai pengguna akar, atau memasukkan ID akun Anda untuk masuk sebagai pengguna IAM. Anda dapat memberikan kata sandi Anda hanya di halaman masuk yang sesuai dengan jenis pengguna Anda. Untuk informasi selengkapnya, lihat [Masuk ke AWS Management Console](#).

Jika Anda berada di halaman masuk yang benar dan kehilangan atau lupa kata sandi atau kunci akses, Anda tidak dapat mengambilnya dari IAM. Sebaliknya, Anda dapat mengatur ulang menggunakan metode berikut:

- Pengguna root akun AWS kata sandi — Jika Anda lupa kata sandi pengguna root Anda, Anda dapat mengatur ulang kata sandi dari AWS Management Console. Untuk detailnya, lihat [the section called “Mengatur ulang kata sandi pengguna akar yang hilang atau terlupakan”](#) nanti dalam topik ini.
- Akun AWS kunci akses — Jika Anda lupa kunci akses akun Anda, Anda dapat membuat kunci akses baru tanpa menonaktifkan kunci akses yang ada. Jika Anda tidak menggunakan tombol yang ada, Anda dapat menghapusnya. Untuk detailnya, lihat [Membuat kunci akses untuk pengguna root](#) dan [Menghapus kunci akses untuk pengguna root](#).
- Kata sandi pengguna IAM – Jika Anda adalah pengguna IAM dan lupa kata sandi, Anda harus meminta administrator untuk mengatur ulang kata sandi Anda. Untuk mempelajari bagaimana administrator dapat mengelola kata sandi Anda, lihat [Mengelola kata sandi untuk pengguna IAM](#).
- Kunci akses pengguna IAM – Jika Anda adalah pengguna IAM dan Anda lupa kunci akses, Anda akan memerlukan kunci akses baru. Jika Anda memiliki izin untuk membuat kunci akses Anda sendiri, Anda dapat menemukan petunjuk untuk membuat kunci akses baru di [Mengelola access key \(konsol\)](#). Jika Anda tidak memiliki izin yang diperlukan, Anda harus meminta administrator Anda untuk membuat access key baru. Jika Anda masih menggunakan kunci lama, minta administrator Anda untuk tidak menghapus kunci lama. Untuk mempelajari bagaimana administrator dapat mengelola access key Anda, lihat [Mengelola kunci akses untuk pengguna IAM](#).

Menggunakan otentikasi multi-faktor (MFA) di AWS

 [Follow us on Twitter](#)

Untuk meningkatkan keamanan, kami menyarankan Anda mengonfigurasi otentikasi multi-faktor (MFA) untuk membantu melindungi sumber daya Anda. AWS Anda dapat mengaktifkan MFA untuk pengguna Pengguna root akun AWS dan IAM. 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.

Anda dapat mendaftarkan hingga delapan perangkat MFA dari kombinasi jenis MFA yang saat ini didukung dengan pengguna Anda Pengguna root akun AWS dan IAM. Untuk informasi selengkapnya tentang jenis MFA yang didukung, lihat [Apa itu MFA?](#) Dengan beberapa perangkat MFA, hanya satu perangkat MFA yang diperlukan untuk masuk ke AWS Management Console atau membuat sesi melalui sebagai pengguna tersebut AWS CLI .

Note

Kami menyarankan Anda meminta pengguna manusia Anda untuk menggunakan kredensial sementara saat mengakses. AWS Sudahkah Anda mempertimbangkan untuk menggunakan AWS IAM Identity Center? Anda dapat menggunakan Pusat Identitas IAM untuk mengelola akses ke beberapa secara terpusat Akun AWS dan memberi pengguna akses masuk tunggal yang dilindungi MFA ke semua akun yang ditetapkan dari satu tempat. Dengan IAM Identity Center, Anda dapat membuat dan mengelola identitas pengguna di IAM Identity Center atau dengan mudah terhubung ke penyedia identitas kompatibel SAMP 2.0 yang ada. Untuk informasi lebih lanjut, lihat [Apa itu Pusat Identitas IAM?](#) dalam AWS IAM Identity Center User Guide.

Apa itu MFA?

MFA menambahkan keamanan ekstra karena mengharuskan pengguna untuk memberikan otentikasi unik dari mekanisme AWS MFA yang didukung selain kredensi masuk reguler mereka ketika mereka mengakses situs web atau layanan. AWS AWS mendukung jenis MFA berikut.

Keamanan FIDO

Kunci keamanan perangkat keras Bersertifikat FIDO disediakan oleh penyedia pihak ketiga.

Aliansi FIDO menyimpan daftar semua [produk Bersertifikat FIDO](#) yang kompatibel dengan spesifikasi FIDO. Standar otentikasi FIDO didasarkan pada kriptografi kunci publik, yang memungkinkan otentikasi yang kuat dan tahan phishing yang lebih aman daripada kata sandi. Kunci keamanan FIDO mendukung beberapa akun root dan pengguna IAM menggunakan satu kunci keamanan. Untuk informasi selengkapnya tentang mengaktifkan kunci keamanan FIDO, lihat [Mengaktifkan kunci keamanan FIDO \(konsol\)](#)

Perangkat MFA virtual

Aplikasi otentikator virtual yang berjalan di ponsel atau perangkat lain dan mengemulasi perangkat fisik.

Aplikasi otentikator virtual menerapkan algoritma [kata sandi satu kali berbasis waktu](#) (TOTP) dan mendukung beberapa token pada satu perangkat. Pengguna harus mengetik kode yang valid dari perangkat di halaman web kedua saat masuk. Setiap perangkat MFA virtual yang ditetapkan ke pengguna harus unik. Pengguna tidak dapat mengetik kode dari perangkat MFA virtual pengguna lain untuk mengautentikasi. Karena mereka dapat berjalan di perangkat seluler yang tidak aman, MFA virtual mungkin tidak memberikan tingkat keamanan yang sama dengan kunci keamanan FIDO.

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 [Autentikasi Multi-Faktor](#). Untuk petunjuk tentang pengaturan perangkat MFA virtual dengan AWS, lihat [Mengaktifkan perangkat Autentikasi Multi-Faktor \(MFA\) virtual \(konsol\)](#)

Token TOTP perangkat keras

Perangkat keras yang 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 keras MFA yang didukung, lihat [Autentikasi Multi-Faktor](#). Untuk petunjuk tentang menyiapkan token TOTP perangkat keras dengan AWS, lihat [Mengaktifkan token TOTP perangkat keras \(konsol\)](#)

Kami menyarankan Anda menggunakan kunci keamanan FIDO sebagai alternatif perangkat TOTP perangkat keras. Kunci keamanan FIDO menawarkan manfaat tanpa persyaratan baterai, ketahanan phishing, dan mereka mendukung beberapa pengguna IAM atau root pada satu perangkat untuk meningkatkan keamanan.

Note

MFA berbasis pesan teks SMS - 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: kunci keamanan FIDO, perangkat MFA virtual \(berbasis perangkat lunak\), 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.

Topik

- [Mengaktifkan perangkat MFA untuk pengguna di AWS](#)
- [Memeriksa status MFA](#)
- [Mensinkronisasi ulang perangkat MFA virtual dan perangkat keras](#)
- [Menonaktifkan perangkat MFA](#)
- [Bagaimana jika perangkat MFA hilang atau berhenti bekerja?](#)
- [Mengonfigurasi akses API yang dilindungi MFA](#)
- [Kode sampel: Meminta kredensial dengan autentikasi multi-faktor](#)

Mengaktifkan perangkat MFA untuk pengguna di AWS

Langkah-langkah untuk mengonfigurasi MFA bergantung pada jenis perangkat MFA yang Anda gunakan.

Topik

- [Langkah umum untuk mengaktifkan perangkat MFA](#)
- [Mengaktifkan perangkat Autentikasi Multi-Faktor \(MFA\) virtual \(konsol\)](#)
- [Mengaktifkan kunci keamanan FIDO \(konsol\)](#)
- [Mengaktifkan token TOTP perangkat keras \(konsol\)](#)
- [Mengaktifkan dan mengelola perangkat MFA virtual \(AWS CLI atau API AWS\)](#)

Langkah umum untuk mengaktifkan perangkat MFA

Prosedur ikhtisar berikut menjelaskan cara menyiapkan dan menggunakan MFA dan menyediakan tautan ke informasi terkait.

Catatan

Anda juga dapat menonton video berbahasa Inggris ini, [Cara Setup AWS Multi-Factor Authentication \(MFA\) Authentication and Budget Alerts](#), untuk informasi lebih lanjut. [AWS](#)

1. Dapatkan perangkat MFA seperti salah satu dari berikut ini. Anda dapat mengaktifkan hingga delapan perangkat MFA per Pengguna root akun AWS atau pengguna IAM dari kombinasi jenis berikut.
 - Perangkat MFA virtual, yang merupakan aplikasi perangkat lunak yang mematuhi [RFC 6238, algoritme TOTP berbasis-standar \(kata sandi satu kali berbasis waktu\)](#). Anda dapat menginstal aplikasi di ponsel atau perangkat lain. Untuk daftar beberapa aplikasi yang didukung yang dapat Anda gunakan sebagai perangkat MFA virtual, lihat [Autentikasi Multi-Faktor](#).
 - Kunci keamanan FIDO dengan [konfigurasi yang AWS didukung](#). Aliansi FIDO menyimpan daftar semua [produk Bersertifikat FIDO](#) yang kompatibel dengan spesifikasi FIDO.
 - Perangkat MFA berbasis perangkat keras dari penyedia pihak ketiga, seperti perangkat token. Token ini digunakan secara eksklusif dengan Akun AWS. Untuk informasi selengkapnya, lihat [Mengaktifkan token TOTP perangkat keras \(konsol\)](#). Anda hanya dapat menggunakan token yang memiliki benih token unik mereka dibagikan dengan AWS aman. 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](#).

2. Aktifkan perangkat MFA.

- Token TOTP Virtual atau Hardware —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 AWS Management Console untuk mengaktifkan perangkat MFA untuk pengguna root.
- Kunci keamanan FIDO — Pengguna root dan pengguna IAM dengan kunci keamanan FIDO dapat mengaktifkan dari AWS Management Console satu-satunya, bukan dari atau API. AWS CLI AWS

Untuk informasi tentang mengaktifkan setiap jenis perangkat MFA, lihat halaman berikut:

- Perangkat MFA virtual: [Mengaktifkan perangkat Autentikasi Multi-Faktor \(MFA\) virtual \(konsol\)](#)
- Kunci keamanan FIDO: [Mengaktifkan kunci keamanan FIDO \(konsol\)](#)
- Token TOTP perangkat keras: [Mengaktifkan token TOTP perangkat keras \(konsol\)](#)

3. Aktifkan Beberapa perangkat MFA (disarankan)

- Kami menyarankan Anda mengaktifkan beberapa perangkat MFA untuk Pengguna root akun AWS dan pengguna IAM di Anda. Akun AWS Ini memungkinkan Anda untuk meningkatkan 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 AWS Management Console atau membuat sesi melalui sebagai pengguna tersebut AWS CLI . Pengguna IAM harus mengautentikasi 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 tanpa melakukan prosedur Akun AWS pemulihan. Akun AWS 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 AWS mengakses tanpa harus mengoordinasikan pertukaran fisik perangkat keras tunggal antara karyawan.
 - Penggunaan perangkat MFA tambahan untuk prinsipal IAM memungkinkan Anda untuk menggunakan satu atau lebih MFA untuk penggunaan sehari-hari, sementara juga menjaga perangkat MFA fisik di lokasi fisik yang aman seperti brankas atau aman untuk cadangan dan redundansi.
4. Gunakan perangkat MFA saat Anda masuk atau mengakses AWS sumber daya. Perhatikan hal berikut:
- Kunci keamanan FIDO — Untuk mengakses AWS situs web, masukkan kredensial Anda, lalu ketuk kunci keamanan FIDO saat diminta.
 - Perangkat MFA virtual dan token TOTP perangkat keras — Untuk mengakses AWS situs web, Anda memerlukan kode MFA dari perangkat selain nama pengguna dan kata sandi Anda.

Untuk mengakses operasi API yang dilindungi MFA, Anda memerlukan hal berikut:

- Kode MFA
- Pengidentifikasi untuk perangkat MFA (nomor seri perangkat fisik atau ARN perangkat virtual yang ditentukan dalam) AWS
- Access key ID dan secret access key biasa

Catatan

- Anda tidak dapat meneruskan informasi MFA untuk kunci keamanan FIDO ke operasi AWS STS API untuk meminta kredensial sementara.

- 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 perangkat root atau IAM MFA.

Lihat informasi yang lebih lengkap di [Menggunakan perangkat MFA dengan halaman masuk IAM Anda](#).

Mengaktifkan perangkat Autentikasi Multi-Faktor (MFA) virtual (konsol)

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 mereka dapat berjalan di perangkat seluler yang tidak aman, MFA virtual mungkin tidak memberikan tingkat keamanan yang sama dengan kunci keamanan FIDO. Kami menyarankan agar Anda menggunakan perangkat MFA virtual saat menunggu untuk mendapatkan persetujuan pembelian perangkat keras atau saat menunggu kedatangan perangkat keras Anda.

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 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 AWS Management Console atau membuat sesi melalui sebagai pengguna tersebut AWS CLI. Kami menyarankan Anda mendaftarkan beberapa perangkat MFA. Untuk aplikasi autentikator, 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 autentikator.

Untuk daftar aplikasi MFA virtual yang dapat Anda gunakan, lihat Otentikasi [Multi-Faktor](#). AWS membutuhkan aplikasi MFA virtual yang menghasilkan OTP enam digit.

Topik

- [Izin diperlukan](#)
- [Aktifkan perangkat MFA virtual untuk pengguna IAM \(konsol\)](#)
- [Ganti 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 AWS Management Console 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 ketika Anda menggunakan API AWS CLI atau AWS. Untuk mengaktifkan dan mengelola perangkat MFA menggunakan API AWS CLI atau AWS, lihat [Mengaktifkan dan mengelola perangkat MFA virtual \(AWS CLI atau API AWS\)](#). Untuk informasi selengkapnya tentang menandai sumber daya IAM, lihat [Menandai sumber daya IAM](#)

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 contoh kebijakan IAM yang memberikan izin ini, lihat kebijakan [Tutorial IAM: Izinkan pengguna untuk mengelola kredensi dan pengaturan MFA mereka](#) dan contoh [AWS: Memungkinkan pengguna IAM yang diautentikasi MFA untuk mengelola perangkat MFA mereka sendiri di halaman kredensi Keamanan](#)

Untuk mengaktifkan perangkat MFA virtual untuk pengguna IAM (konsol)

1. Masuk ke AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Di panel navigasi, pilih Pengguna.
3. Dalam daftar Pengguna, pilih nama pengguna IAM.

4. Pilih tab Kredensial Keamanan. Di bawah Autentikasi multi-faktor (MFA), pilih Tetapkan perangkat MFA.
5. Di wizard, ketikkan nama Perangkat, pilih Aplikasi Authenticator, lalu pilih 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-aplikasi MFA virtual yang dapat Anda gunakan untuk hosting perangkat MFA virtual, lihat [Autentikasi Multi-Faktor](#).

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. Misalnya, Anda dapat memilih ikon kamera atau memilih opsi yang serupa dengan Pindai kode, lalu gunakan 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 kini siap digunakan dengan AWS. Untuk informasi lebih lanjut tentang penggunaan MFA dengan AWS Management Console, lihat [Menggunakan perangkat MFA dengan halaman masuk IAM Anda](#).

Ganti perangkat MFA virtual

Anda dapat mendaftarkan hingga delapan perangkat MFA dari kombinasi [jenis MFA yang saat ini didukung 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 IAM lain, lihat [Menonaktifkan 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 prosedur. [Aktifkan perangkat MFA virtual untuk Pengguna root akun AWS \(konsol\) Anda](#)

Mengaktifkan kunci keamanan FIDO (konsol)

Kunci keamanan FIDO adalah jenis [perangkat otentikasi multi-faktor \(MFA\)](#) yang dapat Anda gunakan untuk melindungi sumber daya Anda. AWS Anda mencolokkan kunci keamanan FIDO Anda ke port USB di komputer Anda dan mengaktifkannya menggunakan instruksi yang mengikuti. Setelah Anda mengaktifkannya, Anda mengetuknya ketika 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 AWS MFA di. 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 [Autentikasi Multi-Faktor](#). Untuk spesifikasi dan informasi pembelian, lihat [Autentikasi Multi-Faktor](#).

FIDO2 adalah standar otentikasi terbuka dan perpanjangan dari FIDO U2F, menawarkan tingkat keamanan yang sama tinggi 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 autentikator eksternal. Saat Anda mengaktifkan autentikator Bersertifikat FIDO AWS, kunci keamanan FIDO membuat key pair baru untuk digunakan dengan saja. AWS Pertama, Anda memasukkan kredensial Anda. Saat diminta, Anda mengetuk kunci keamanan FIDO, yang merespons tantangan otentikasi yang dikeluarkan oleh. AWS Untuk mempelajari lebih lanjut tentang standar FIDO2, lihat Proyek [FIDO2](#).

Anda dapat mendaftarkan hingga delapan perangkat MFA dari kombinasi [jenis MFA yang saat ini didukung dengan pengguna Akun AWS root dan pengguna IAM](#) Anda. Dengan beberapa perangkat MFA, Anda hanya perlu satu perangkat MFA untuk masuk ke AWS Management Console atau membuat sesi melalui sebagai pengguna tersebut AWS CLI . 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 Anda, maka Anda dapat menggunakan kunci keamanan terdaftar Anda. Untuk aplikasi autentikator, 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 autentikator.

Note

Kami menyarankan Anda meminta pengguna manusia Anda untuk menggunakan kredensi sementara saat mengakses. AWS Pengguna Anda dapat bergabung AWS dengan penyedia identitas tempat mereka mengautentikasi dengan kredensi perusahaan dan konfigurasi MFA mereka. Untuk mengelola akses ke AWS dan aplikasi bisnis, kami sarankan Anda menggunakan IAM Identity Center. Untuk informasi selengkapnya, lihat [Panduan Pengguna Pusat Identitas IAM](#).

Topik

- [Izin diperlukan](#)
- [Aktifkan kunci keamanan FIDO untuk pengguna IAM Anda sendiri \(konsol\)](#)
- [Aktifkan kunci keamanan FIDO untuk pengguna IAM lain \(konsol\)](#)
- [Ganti kunci keamanan FIDO](#)
- [Konfigurasi yang didukung untuk menggunakan kunci keamanan FIDO](#)

Izin diperlukan

Untuk mengelola kunci keamanan FIDO untuk pengguna IAM Anda sendiri sambil melindungi tindakan sensitif terkait MFA, Anda harus memiliki izin dari kebijakan berikut:

Note

Nilai ARN adalah nilai statis dan bukan merupakan indikator protokol apa yang digunakan untuk mendaftarkan autentikator. Kami telah menghentikan U2F, jadi semua implementasi baru digunakan. WebAuthn

```
{
  "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"
        }
      }
    }
  ]
}
```


```
}  
  ]  
}
```

Aktifkan kunci keamanan FIDO untuk pengguna IAM Anda sendiri (konsol)

Anda dapat mengaktifkan kunci keamanan FIDO untuk pengguna IAM Anda sendiri dari AWS Management Console satu-satunya, bukan dari AWS CLI atau AWS API.

 Note


Sebelum Anda dapat mengaktifkan kunci keamanan FIDO, Anda harus memiliki akses fisik ke perangkat.

 Note

Anda tidak boleh memilih salah satu opsi yang tersedia di pop-up Google Chrome yang meminta Verifikasi identitas Anda dengan amazon.com. Anda hanya perlu mengetuk tombol keamanan.

Untuk mengaktifkan kunci keamanan FIDO untuk pengguna IAM Anda sendiri (konsol)

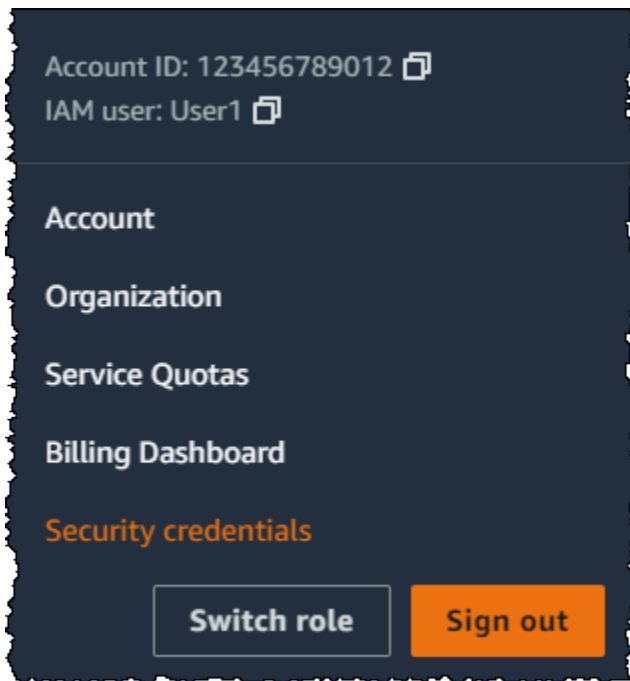
1. Gunakan ID AWS akun atau alias akun, nama pengguna IAM, dan kata sandi Anda untuk masuk ke konsol [IAM](#).

 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 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, di bagian Otentikasi Multi-faktor (MFA), pilih Tetapkan perangkat MFA.
4. Di wizard, ketikkan nama Perangkat, pilih Kunci Keamanan, lalu pilih Berikutnya.
5. Masukkan kunci keamanan FIDO ke port USB komputer Anda.



6. Ketuk tombol keamanan FIDO.

Kunci keamanan FIDO siap digunakan. AWS Untuk informasi tentang menggunakan MFA dengan AWS Management Console, lihat. [Menggunakan perangkat MFA dengan halaman masuk IAM Anda](#)

Aktifkan kunci keamanan FIDO untuk pengguna IAM lain (konsol)

Anda dapat mengaktifkan kunci keamanan FIDO untuk pengguna IAM lain dari AWS Management Console satu-satunya, bukan dari AWS CLI atau AWS API.

Untuk mengaktifkan kunci keamanan FIDO untuk pengguna IAM lain (konsol)

1. Masuk ke AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Di panel navigasi, pilih Pengguna.
3. Pilih nama pengguna yang ingin Anda aktifkan MFA.
4. Pilih tab Kredensial Keamanan. Di bawah Autentikasi multi-faktor (MFA), pilih Tetapkan perangkat MFA.
5. Di wizard, ketikkan nama Perangkat, pilih Kunci Keamanan, lalu pilih Berikutnya.
6. Masukkan kunci keamanan FIDO ke port USB komputer Anda.



7. Ketuk tombol keamanan FIDO.

Kunci keamanan FIDO siap digunakan. AWS Untuk informasi tentang menggunakan MFA dengan AWS Management Console, lihat. [Menggunakan perangkat MFA dengan halaman masuk IAM Anda](#)

Ganti kunci keamanan FIDO

Anda dapat memiliki hingga delapan perangkat MFA dari kombinasi [jenis MFA yang saat ini didukung](#) yang ditetapkan untuk digunakan 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 autentikator FIDO lama. Kemudian Anda dapat menambahkan perangkat MFA baru untuk pengguna.

- Untuk menonaktifkan perangkat yang saat ini terkait dengan pengguna IAM, lihat. [Menonaktifkan perangkat MFA](#)
- Untuk menambahkan kunci keamanan FIDO baru untuk pengguna IAM, lihat. [Aktifkan kunci keamanan FIDO untuk pengguna IAM Anda sendiri \(konsol\)](#)

Jika Anda tidak memiliki akses ke kunci keamanan FIDO baru, Anda dapat mengaktifkan perangkat MFA virtual baru atau token TOTP perangkat keras. Lihat salah satu dari yang berikut untuk petunjuk:

- [Mengaktifkan perangkat Autentikasi Multi-Faktor \(MFA\) virtual \(konsol\)](#)
- [Mengaktifkan token TOTP perangkat keras \(konsol\)](#)

Konfigurasi yang didukung untuk menggunakan kunci keamanan FIDO

Anda dapat menggunakan kunci keamanan FIDO2 sebagai metode otentikasi multi-faktor (MFA) dengan IAM menggunakan konfigurasi yang didukung saat ini. Ini termasuk perangkat FIDO2 yang didukung oleh IAM dan browser yang mendukung FIDO2. Sebelum Anda mendaftarkan perangkat FIDO2 Anda, periksa apakah Anda menggunakan versi browser dan sistem operasi (OS) terbaru. Fitur dapat berperilaku berbeda di berbagai browser, autentikator, dan klien OS. Jika pendaftaran perangkat Anda gagal pada satu browser, Anda dapat mencoba mendaftar dengan browser lain.

Perangkat FIDO2 didukung oleh AWS

IAM mendukung perangkat keamanan FIDO2 yang terhubung ke perangkat Anda melalui USB, Bluetooth, atau NFC. Kami tidak mendukung otentikator platform seperti TouchID, FaceID, atau Windows Hello.

Note

AWS memerlukan akses ke port USB fisik di komputer Anda untuk memverifikasi perangkat FIDO2 Anda. Kunci keamanan FIDO2 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 FIDO2:

	macOS 10.15+	Windows 10	Linux	iOS 14.5+	Android 7+
Chrome	Ya	Ya	Ya	Ya	Tidak
Safari	Ya	Tidak	Tidak	Ya	Tidak
Tepi	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 [Mengonfigurasi tombol keamanan FIDO](#)

Untuk informasi selengkapnya tentang dukungan browser untuk perangkat bersertifikat FIDO2 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 FIDO. Sertifikasi perangkat Anda diambil dari [Layanan Metadata Aliansi FIDO](#) (MDS). Jika status sertifikasi atau tingkat kunci keamanan FIDO 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 FIDO MDS: FIPS-140-2, FIPS-140-3, dan tingkat sertifikasi FIDO. Anda memiliki kemampuan untuk menentukan pendaftaran autentikator tertentu dalam kebijakan IAM mereka, berdasarkan jenis dan tingkat sertifikasi pilihan Anda. Untuk informasi selengkapnya, 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 mendaftarkan 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 mendaftarkan hanya perangkat yang memiliki sertifikasi FIPS-140-2 L2

```
{
  "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

```

{
  "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

```

{

```

```

"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

```

{
  "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:FIPS-140-2-certification": "L2"
      }
    }
  },
  {
    "Effect": "Allow",
    "Action": "iam:EnableMFADevice",
    "Resource": "*",
    "Condition": {
      "Null": {
        "iam:RegisterSecurityKey": "true"
      }
    }
  }
]
}

```

AWS CLI dan AWS API

AWS mendukung penggunaan kunci keamanan FIDO2 hanya di AWS Management Console [Menggunakan kunci keamanan FIDO2 untuk MFA tidak didukung di AWS API AWS CLI dan, atau untuk akses ke operasi API yang dilindungi MFA.](#)

Sumber daya tambahan

- Untuk informasi selengkapnya tentang penggunaan kunci keamanan FIDO2 AWS, lihat. [Mengaktifkan kunci keamanan FIDO \(konsol\)](#)

- Untuk bantuan dalam memecahkan masalah kunci keamanan FIDO2, lihat. AWS [Mengonfigurasi tombol keamanan FIDO](#)
- Untuk informasi industri umum tentang dukungan FIDO2, lihat Proyek [FIDO2](#).

Mengaktifkan token TOTP perangkat keras (konsol)

Token TOTP perangkat keras menghasilkan kode numerik enam digit berdasarkan algoritma kata sandi satu kali berbasis waktu (TOTP). 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](#) adalah perangkat fisik yang Anda beli. Perangkat MFA perangkat keras menghasilkan kode TOTP untuk otentikasi saat Anda masuk. AWS Mereka mengandalkan baterai, yang mungkin perlu penggantian dan sinkronisasi ulang AWS seiring waktu. Kunci keamanan FIDO, yang menggunakan kriptografi kunci publik, tidak memerlukan baterai dan menawarkan proses otentikasi yang mulus. Kami merekomendasikan penggunaan kunci keamanan FIDO untuk ketahanan phishing mereka, yang memberikan 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 [Autentikasi Multi-Faktor](#).

Anda dapat mengaktifkan token TOTP perangkat keras untuk pengguna IAM dari, baris perintah AWS Management Console, atau API IAM. Untuk mengaktifkan perangkat MFA untuk Anda Pengguna root akun AWS, lihat. [Aktifkan token TOTP perangkat keras untuk pengguna Akun AWS 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 AWS Management Console atau membuat sesi melalui sebagai pengguna tersebut AWS CLI .

Important

Kami menyarankan Anda mengaktifkan beberapa perangkat MFA untuk pengguna Anda untuk melanjutkan akses 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 API IAM, gunakan operasi [EnableMFADevice](#).

Topik

- [Izin diperlukan](#)
- [Aktifkan token TOTP perangkat keras untuk pengguna IAM Anda sendiri \(konsol\)](#)
- [Aktifkan token TOTP perangkat keras untuk pengguna IAM lain \(konsol\)](#)
- [Ganti perangkat MFA fisik](#)

Izin diperlukan

Untuk mengelola token TOTP perangkat keras untuk pengguna IAM Anda sendiri sambil melindungi tindakan sensitif terkait MFA, Anda harus memiliki izin dari kebijakan berikut:


```
{
  "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 AWS Management Console

 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)

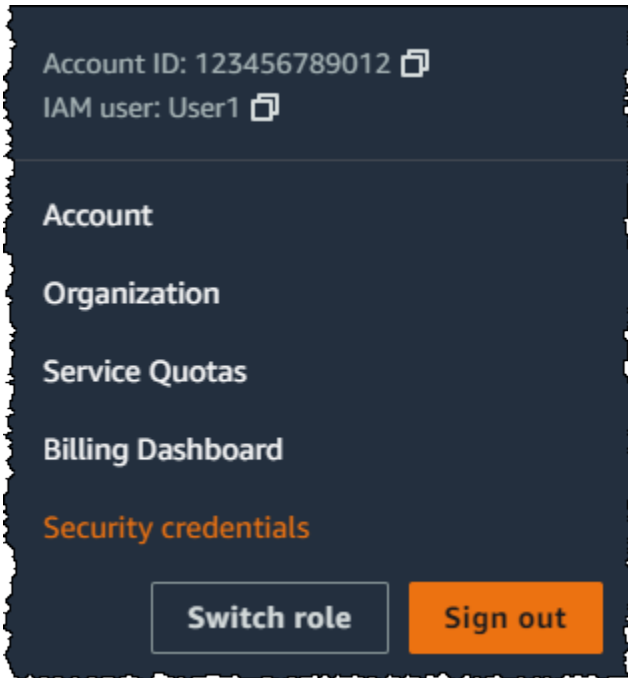
1. Gunakan ID AWS akun atau alias akun, nama pengguna IAM, dan kata sandi Anda untuk masuk ke konsol [IAM](#).

 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 Anda untuk diarahkan ke halaman login pengguna IAM untuk akun Anda.

Untuk mendapatkan Akun AWS ID Anda, hubungi administrator Anda.

- Di bilah navigasi di kanan atas, pilih nama pengguna Anda, lalu pilih Kredensi keamanan.



- Pada tab Kredensial AWS IAM, di bagian Otentikasi Multi-faktor (MFA), pilih Tetapkan perangkat MFA.
- Di wizard, ketikkan nama Perangkat, pilih token TOTP Perangkat Keras, lalu pilih Berikutnya.
- Ketik nomor seri perangkat. Nomor seri biasanya ada di bagian belakang perangkat.
- 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.



- 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.
- 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 AWS Management Console, lihat [Menggunakan perangkat MFA dengan halaman masuk IAM Anda](#).

Aktifkan token TOTP perangkat keras untuk pengguna IAM lain (konsol)

Anda dapat mengaktifkan token TOTP perangkat keras untuk pengguna IAM lain dari AWS Management Console

Untuk mengaktifkan token TOTP perangkat keras untuk pengguna IAM lain (konsol)

1. Masuk ke AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Di panel navigasi, pilih Pengguna.
3. Pilih nama pengguna yang ingin Anda aktifkan MFA.
4. Pilih tab Kredensial Keamanan. Di bawah Autentikasi multi-faktor (MFA), pilih Tetapkan perangkat MFA.
5. Di wizard, ketikkan nama Perangkat, pilih token TOTP Perangkat Keras, lalu pilih Berikutnya.
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.

lainnya. Pengguna root akun AWS Namun, Anda dapat menggunakan AWS Management Console untuk mengaktifkan aktifkan aktifkan MFA untuk pengguna akar.

Saat Anda mengaktifkan perangkat MFA dari AWS Management Console, konsol akan melakukan beberapa langkah untuk Anda. Jika Anda membuat perangkat virtual menggunakan AWS CLI, Tools for Windows PowerShell, atau AWS API, maka Anda harus melakukan langkah-langkah secara manual dan dengan 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 dari [New-IAMVirtualMFADevice](#) untuk detail lebih lanjut. 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 API AWS CLI atau AWS.

Pengguna IAM yang menggunakan SDK atau CLI dapat mengaktifkan perangkat MFA tambahan dengan menelepon [EnableMFADevice](#) atau menonaktifkan perangkat MFA yang ada dengan menelepon [DeactivateMFADevice](#). Untuk melakukan ini dengan sukses, mereka harus terlebih dahulu menelepon [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 [kredendenalGetSessionToken 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](#)
- API AWS: [CreateVirtualMFADevice](#)

Untuk mengaktifkan perangkat MFA untuk digunakan dengan AWS

Perintah ini menyinkronkan 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](#)
- API AWS: [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](#)
- API AWS: [DeactivateMFADevice](#)

Untuk mencantumkan entitas perangkat MFA virtual

Gunakan perintah ini untuk mencantumkan entitas perangkat MFA virtual.

- AWS CLI: [aws iam list-virtual-mfa-devices](#)
- API AWS: [ListVirtualMFADevices](#)

Untuk menandai perangkat MFA virtual

Gunakan perintah ini untuk menandai perangkat MFA virtual.


- AWS CLI: [aws iam tag-mfa-device](#)
- API AWS: [TagMFADevice](#)

Untuk mencantumkan tanda perangkat MFA virtual

Memeriksa status MFA

Gunakan konsol IAM untuk memeriksa apakah perangkat MFA valid yang diaktifkan. Pengguna root akun AWS


Untuk memeriksa status MFA dari pengguna akar

1. Masuklah keAWS Management Console dengan kredensi pengguna akar Anda dan kemudian buka konsol IAM di<https://console.aws.amazon.com/iam/>.
2. Di bilah navigasi di kanan atas, pilih nama pengguna Anda.
3. Periksa di bawah Autentikasi Multi-Faktor (MFA) untuk melihat apakah MFA diaktifkan atau dinonaktifkan. Jika MFA belum diaktifkan, simbol peringatan () akan ditampilkan

Jika Anda ingin mengaktifkan MFA untuk akun, lihat salah satu hal berikut:

- [Aktifkan perangkat MFA virtual untuk Pengguna root akun AWS \(konsol\) Anda](#)
- [Aktifkan kunci keamanan FIDO untuk pengguna Akun AWS root \(konsol\)](#)
- [Aktifkan token TOTP perangkat keras untuk pengguna Akun AWS 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 yang tidak ingin Anda tampilkan dalam tabel pengguna.
 - d. Pilih Tutup untuk kembali ke daftar pengguna.

4. Kolom MFA memberi tahu Anda tentang perangkat MFA yang diaktifkan. Jika tidak ada perangkat MFA yang aktif untuk pengguna, konsol akan menampilkan Tidak ada. Jika pengguna mengaktifkan perangkat MFA, kolom MFA menampilkan jenis perangkat yang diaktifkan dengan nilai Virtual, Kunci Keamanan, Perangkat Keras, atau SMS.

 Note

AWS mengakhiri dukungan untuk mengaktifkan Autentikasi Multi-Faktor (MFA). [Kami merekomendasikan pelanggan yang memiliki pengguna IAM yang menggunakan MFA berbasis pesan teks, atau 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 akan menampilkan Tidak ada perangkat MFA. Tetapkan perangkat MFA untuk meningkatkan keamanan AWS lingkungan Anda di bagian Multi-factor authentication (MFA). Jika pengguna mengaktifkan perangkat MFA, bagian Otentikasi Multi-faktor (MFA) menampilkan detail tentang perangkat:
 - Nama perangkat akar:.....
 - Tipe perangkat akar:
 - Pengidentifikasi untuk perangkat, seperti nomor seri untuk perangkat fisik atau ARN di AWS untuk perangkat virtual
 - Saat perangkat dibuat

Untuk menghapus atau menyinkronkan ulang perangkat, pilih tombol radio di samping perangkat, lalu pilih Hapus atau Sinkronkan Ulang.

Untuk informasi lebih lanjut dalam mengaktifkan MFA, lihat hal berikut:

- [Mengaktifkan perangkat Autentikasi Multi-Faktor \(MFA\) virtual \(konsol\)](#)
- [Mengaktifkan kunci keamanan FIDO \(konsol\)](#)
- [Mengaktifkan token TOTP perangkat keras \(konsol\)](#)

Mengsinkronisasi ulang perangkat MFA virtual dan perangkat keras

Anda dapat menggunakan AWS untuk menyinkronkan ulang perangkat autentikasi multi-faktor (MFA) virtual dan perangkat keras. 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 keluar dari 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 Anda jika mereka keluar dari sinkronisasi.

Jika perangkat Pengguna root akun AWS MFA 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 kaitannya dan mengaitkannya kembali. Untuk informasi lebih lanjut tentang cara melakukan ini, lihat [Menonaktifkan perangkat MFA](#) dan [Mengaktifkan perangkat MFA untuk pengguna di AWS](#).

Topik

- [Izin diperlukan](#)
- [Mensinkronisasi ulang perangkat MFA virtual dan perangkat keras \(konsol IAM\)](#)
- [Menyinkronkan ulang perangkat MFA virtual dan perangkat keras \(AWS CLI\)](#)
- [Menyinkronkan ulang perangkat MFA virtual dan perangkat keras \(API AWS\)](#)

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.

```
{  
  "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 mensinkronisasi ulang perangkat MFA virtual dan perangkat keras.

Untuk mensinkronkan perangkat keras atau MFA virtual untuk pengguna IAM Anda sendiri (konsol)

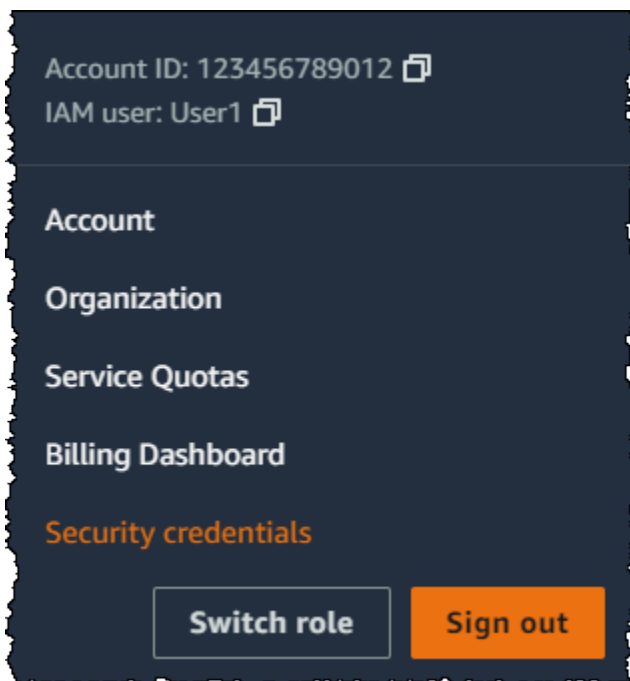
1. Gunakan ID AWS akun atau alias akun, nama pengguna IAM, dan kata sandi Anda untuk masuk ke konsol [IAM](#).

Note

Untuk kenyamanan Anda, halaman masuk AWS menggunakan cookie browser untuk mengingat nama pengguna dan informasi akun pengguna IAM. 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 menyetikkan 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 kredensi AWS IAM, di bagian Otentikasi Multi-faktor (MFA), pilih tombol radio di sebelah perangkat MFA dan pilih Resinkronisasi.
4. Ketik dua kode berikutnya yang dibuat secara berurutan dari perangkat ke dalam Kode MFA 1 dan Kode MFA 2. Kemudian pilih Resinkronisasi.

⚠ 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 AWS Management Console 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 Otentikasi Multi-faktor (MFA), pilih tombol radio di sebelah perangkat MFA dan pilih Resinkronisasi.
4. Ketik dua kode berikutnya yang dibuat secara berurutan dari perangkat ke dalam Kode MFA 1 dan Kode MFA 2. Kemudian pilih Resinkronisasi.

⚠ 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 Sinkronkan ulang Dengan Server Kami, ketikkan dua kode berikutnya yang dibuat secara berurutan dari perangkat ke dalam Kode MFA 1 dan Kode MFA 2. Lalu, pilih Sinkronkan ulang perangkat autentikasi.
3. Jika perlu, ketikkan kata sandi Anda kembali dan pilih Masuk. Lalu selesaikan proses masuk menggunakan perangkat MFA Anda.

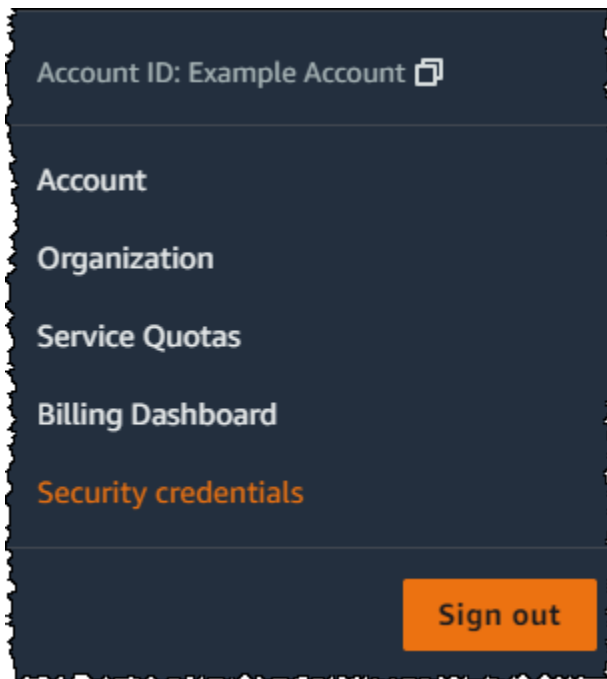
Untuk mensinkronkan ulang MFA pengguna akar Anda setelah masuk (konsol)

1. Masuk ke [Konsol IAM](#) sebagai pemilik akun dengan memilih Pengguna akar dan masukkan alamat email Akun AWS Anda. Di laman berikutnya, masukkan kata sandi Anda.

 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 AWS Management Console 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. Perluas bagian Multi-Factor Authentication (MFA) di halaman.
4. Pilih tombol radio di sebelah perangkat dan pilih Resinkronisasi.
5. Di kotak dialog perangkat MFA Resinkronisasi, ketik dua kode berikutnya yang dihasilkan secara berurutan dari perangkat ke dalam kode MFA 1 dan kode MFA 2. Kemudian pilih Resinkronisasi.

⚠ Important

Kirimkan permintaan Anda segera setelah membuat kode. Jika Anda membuat kode lalu menunggu terlalu lama untuk membuat permintaan, perangkat MFA berhasil dikaitkan dengan pengguna, tetapi perangkat MFA tidak sinkron. Hal ini terjadi karena kata sandi sekali pakai berbasis-waktu kedaluwarsa setelah beberapa waktu.

Menyinkronkan 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

Pada prompt perintah, keluarkan 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.

Menyinkronkan ulang perangkat MFA virtual dan perangkat keras (API AWS)

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 (API AWS)

- Mengirim permintaan [ResyncMFADevice](#).

Menonaktifkan 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 [the section called “Mengaktifkan perangkat MFA”](#).

Note

Jika Anda menggunakan API atau AWS CLI menghapus pengguna dari AndaAkun AWS, Anda harus menonaktifkan atau menghapus perangkat MFA pengguna. Anda membuat perubahan ini sebagai bagian dari proses penghapusan pengguna. Untuk informasi lebih lanjut tentang penghapusan pengguna, lihat [Mengelola pengguna IAM](#).

Topik

- [Menonaktifkan perangkat MFA \(konsol\)](#)
- [Menonaktifkan perangkat MFA \(AWS CLI\)](#)
- [Menonaktifkan perangkat MFA \(API AWS\)](#)

Menonaktifkan perangkat MFA (konsol)

Untuk menonaktifkan perangkat MFA untuk pengguna IAM lainnya (konsol)

1. Masuk ke AWS Management Console 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 MFAnyanya ingin Anda hapus.
4. Pilih tab Kredensial keamanan.
5. Di bawah Autentikasi multi-faktor (MFA), pilih tombol radio di sebelah perangkat MFA, pilih Hapus, lalu pilih Hapus.

Perangkat dihapus dari AWS. Perangkat tidak dapat digunakan untuk masuk atau mengautentikasi permintaan hingga diaktivasi ulang dan dikaitkan dengan pengguna AWS atau Pengguna root akun AWS.

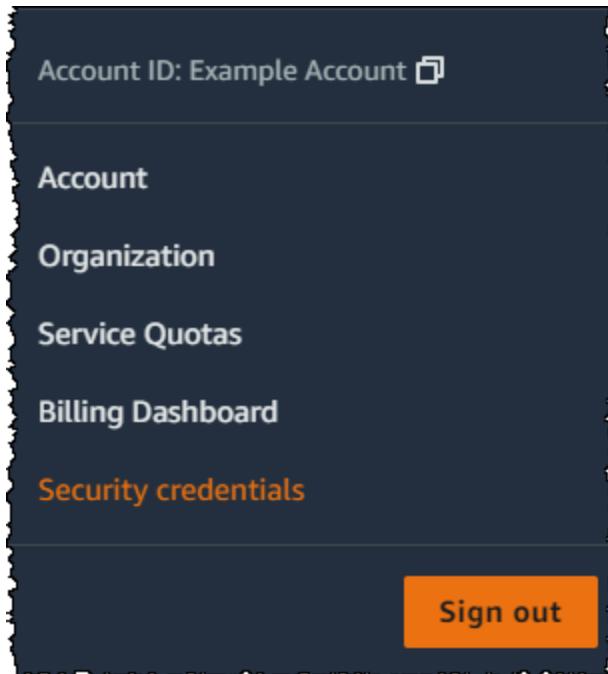
Untuk menonaktifkan perangkat MFA bagi Pengguna root akun AWS Anda (konsol)

1. Masuk ke [Konsol IAM](#) sebagai pemilik akun dengan memilih Pengguna akar dan masukkan alamat email Akun AWS Anda. Di laman berikutnya, masukkan kata sandi Anda.

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 AWS Management Console 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 Otentikasi multi-faktor (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.

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 (API AWS)

Untuk menonaktifkan perangkat MFA untuk pengguna IAM (API) AWS

- Hubungi operasi ini: [DeactivateMFADevice](#)

Bagaimana jika perangkat MFA hilang atau berhenti bekerja?

Jika perangkat [MFA virtual atau token TOTP perangkat keras](#) Anda tampaknya berfungsi dengan baik, tetapi Anda tidak dapat menggunakannya untuk mengakses AWS sumber daya Anda, itu mungkin tidak sinkronisasi dengan. AWS Untuk informasi tentang sinkronisasi perangkat MFA virtual atau MFA perangkat keras, lihat [Mengsinkronisasi ulang perangkat MFA virtual dan perangkat keras](#). [Kunci keamanan FIDO](#) tidak keluar dari sinkron.

Jika [perangkat otentikasi Pengguna root akun AWS multi-faktor \(MFA\)](#) Anda hilang, rusak, atau tidak berfungsi, Anda dapat memulihkan akses ke akun Anda. Pengguna IAM harus menghubungi administrator untuk menonaktifkan perangkat.

Important

Kami menyarankan Anda mengaktifkan beberapa perangkat MFA untuk pengguna IAM Anda untuk memastikan akses berkelanjutan ke akun Anda jika perangkat MFA hilang atau tidak dapat diakses. Anda dapat mendaftarkan hingga delapan perangkat MFA dari kombinasi jenis MFA yang saat ini didukung dengan pengguna Akun AWS root dan pengguna IAM Anda.

Pemulihan perangkat MFA pengguna akar

Jika perangkat [otentikasi Pengguna root akun AWS multi-faktor \(MFA\) Anda hilang, rusak, atau tidak berfungsi](#), Anda dapat masuk menggunakan perangkat MFA lain yang terdaftar di perangkat yang sama. Pengguna root akun AWS Jika pengguna root hanya memiliki satu perangkat MFA yang diaktifkan, Anda dapat menggunakan metode otentikasi alternatif. Ini berarti bahwa jika Anda tidak dapat masuk dengan perangkat MFA Anda, Anda dapat masuk dengan memverifikasi identitas Anda menggunakan email dan nomor telepon kontak utama yang terdaftar di akun Anda.

Sebelum Anda menggunakan faktor otentikasi alternatif untuk masuk sebagai pengguna root, Anda harus dapat mengakses email dan nomor telepon kontak utama yang terkait dengan akun Anda. Jika Anda perlu memperbarui nomor telepon kontak utama, Anda dapat masuk sebagai pengguna IAM dengan akses Administrator, bukan 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 Support](#).

Important

Kami menyarankan agar Anda tetap memperbarui alamat email dan nomor telepon kontak ke pengguna root Anda agar pemulihan akun berhasil. Untuk informasi selengkapnya, lihat [Memperbarui kontak utama untuk Anda Akun AWS](#) di Panduan AWS Account Management Referensi.

Untuk masuk menggunakan faktor autentikasi alternatif sebagai Pengguna root akun AWS

1. Masuk ke [AWS Management Console](#) sebagai pemilik akun dengan memilih pengguna Root dan memasukkan alamat Akun AWS email Anda. Di laman berikutnya, masukkan kata sandi Anda.
2. Pada halaman Verifikasi tambahan yang diperlukan, pilih metode MFA untuk mengautentikasi dan pilih Berikutnya.

Note

Anda mungkin melihat teks alternatif, seperti Masuk menggunakan MFA, Memecahkan masalah perangkat otentikasi Anda, atau Memecahkan 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 Support](#) untuk menonaktifkan perangkat MFA Anda.

3. Bergantung pada jenis MFA yang Anda gunakan, Anda akan melihat halaman yang berbeda, tetapi opsi Pemecahan Masalah MFA berfungsi sama. Pada halaman verifikasi tambahan yang diperlukan atau halaman otentikasi multi-faktor, pilih Troubleshoot MFA.
4. Jika perlu, ketikkan kata sandi Anda kembali dan pilih Masuk.
5. Pada halaman Memecahkan masalah perangkat autentikasi Anda, di bagian Masuk menggunakan faktor alternatif autentikasi, pilih Masuk menggunakan faktor alternatif.
6. Pada halaman Masuk menggunakan faktor alternatif otentikasi, 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 Hubungi saya sekarang.
9. Jawab panggilan dari AWS dan, jika diminta, masukkan 6-digit angka dari situs AWS pada keypad ponsel Anda.


Jika Anda tidak menerima panggilan dari AWS, pilih Masuk untuk masuk ke konsol dan memulai kembali. Atau lihat Perangkat [Multi-Factor Authentication \(MFA\) yang hilang atau tidak dapat digunakan untuk menghubungi dukungan untuk mendapatkan 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, buka halaman [AWS Security Credentials](#) 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 MFA

perangkat keras, kunjungi halaman [Kredensial Keamanan AWS](#) dan hapus perangkat virtual MFA lama sebelum Anda membuat perangkat baru.

 Note

Anda tidak perlu mengganti perangkat MFA yang hilang atau dicuri dengan jenis perangkat yang sama. Misalnya, jika Anda memecahkan kunci keamanan FIDO dan memesan yang baru, Anda dapat menggunakan MFA virtual atau token TOTP perangkat keras hingga Anda menerima kunci keamanan FIDO baru.

 Important

Jika perangkat MFA Anda hilang atau dicuri, setelah masuk menggunakan faktor otentikasi alternatif dan membuat perangkat MFA pengganti Anda, ubah kata sandi pengguna root Anda jika penyerang telah mencuri perangkat otentikasi dan mungkin juga memiliki kata sandi Anda saat ini. Untuk informasi selengkapnya, lihat [Mengubah kata sandi untuk Pengguna root akun AWS di](#) Panduan AWS Account Management Referensi.

Pemulihan perangkat MFA pengguna IAM

Jika Anda adalah pengguna IAM dan perangkat Anda hilang atau berhenti berfungsi, Anda tidak dapat memulihkannya sendiri. Anda harus menghubungi administrator untuk menonaktifkan perangkat. Kemudian Anda dapat mengaktifkan perangkat baru.

Untuk mendapatkan bantuan bagi perangkat MFA sebagai pengguna IAM

1. Hubungi administrator AWS atau orang lain yang memberikan nama pengguna dan kata sandi untuk pengguna IAM. Administrator harus menonaktifkan perangkat MFA seperti yang dijelaskan dalam [Menonaktifkan 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 [Mengaktifkan perangkat Autentikasi Multi-Faktor \(MFA\) virtual \(konsol\)](#).

- Untuk kunci keamanan FIDO, hubungi penyedia pihak ketiga untuk bantuan mengganti perangkat. Saat Anda menerima kunci keamanan FIDO baru, aktifkan seperti yang dijelaskan di [Mengaktifkan kunci keamanan FIDO \(konsol\)](#).
- 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 [Mengaktifkan token TOTP perangkat keras \(konsol\)](#).

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 dan memesan yang baru, Anda dapat menggunakan MFA virtual atau token TOTP perangkat keras hingga Anda menerima kunci keamanan FIDO baru.

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 [Mengelola kata sandi untuk pengguna IAM](#)

Mengonfigurasi akses API yang dilindungi MFA

Dengan kebijakan IAM, Anda dapat menentukan operasi API mana yang diizinkan untuk dipanggil oleh pengguna. Dalam beberapa kasus, Anda mungkin ingin tambahan keamanan yang mewajibkan pengguna untuk diautentikasi dengan autentikasi multi-faktor (MFA) AWS 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 merusak seperti `TerminateInstances` dan memastikan bahwa pengguna dapat melakukan tindakan tersebut hanya jika mereka mengautentikasi dengan perangkat AWS.

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 MFA AWS untuk setiap pengguna yang perlu membuat permintaan API yang memerlukan autentikasi MFA. Proses ini dijelaskan di [Mengaktifkan perangkat MFA untuk pengguna di AWS](#).
2. Administrator membuat kebijakan untuk pengguna yang mencakup elemen `Condition` yang memeriksa apakah pengguna mengautentikasi dengan perangkat MFA AWS.
3. Pengguna memanggil salah satu operasi AWS STS API yang mendukung parameter MFA [AssumeRole](#) atau [GetSessionToken](#), bergantung pada skenario perlindungan MFA, seperti yang akan dijelaskan nanti. 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 yang ditolak (seperti yang dilakukan terhadap akses tanpa izin). Dengan kebijakan API yang dilindungi MFA, AWS menolak akses ke operasi API yang ditentukan dalam kebijakan jika pengguna mencoba memanggil operasi API tanpa autentikasi 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`. Kunci tersebut hanya muncul ketika pengguna mengautentikasi dengan kredensial jangka pendek. Kredensial jangka panjang, seperti access key, tidak mencakup 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` 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 elemen `Akun AWS` yang ditentukan dalam `Principal` elemen (ganti `ACCOUNT-B-ID` dengan `Akun AWS ID` yang valid) dapat mengambil peran yang terkait dengan kebijakan ini. Namun, pengguna tersebut hanya dapat mengasumsikan peran jika pengguna diautentikasi menggunakan MFA.

```
{
  "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 menyampaikan 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:

- Hubungi operasi yang mengakses sumber daya dalam operasi yang Akun AWS sama dengan pengguna IAM yang mengajukan permintaan. Perhatikan bahwa kredensial sementara dari `GetSessionToken` permintaan dapat mengakses operasi operasi 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:

- Hubungi operasi yang mengakses sumber daya dalam operasi API yang sama atau berbeda Akun AWS. Panggilan API dapat mencakup API IAM atau panggilan API dapat mencakup API IAM atau AWS STS API dapat mencakup API IAM atau panggilan. 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.

Perincian tentang cara menerapkan skenario ini akan dijelaskan nanti dalam dokumen ini.

Poin penting tentang akses API yang dilindungi MFA

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 API yang dilindungi MFA dengan kredensial Pengguna root akun AWS.
- Anda tidak dapat menggunakan akses API yang dilindungi MFA dengan kunci keamanan U2F.
- Pengguna federasi tidak dapat diberi perangkat MFA untuk digunakan dengan layanan AWS, sehingga mereka tidak dapat mengakses sumber daya AWS yang dikontrol oleh MFA. (Lihat poin berikutnya.)
- Operasi API AWS STS lainnya yang mengembalikan kredensial sementara tidak mendukung MFA. Untuk `AssumeRoleWithWebIdentity` dan `AssumeRoleWithSAML`, pengguna diautentikasi oleh penyedia eksternal dan AWS tidak dapat menentukan apakah penyedia tersebut membutuhkan MFA. Untuk `GetFederationToken`, MFA tidak selalu berkaitan dengan pengguna tertentu.
- Demikian pula, kredensial jangka panjang (kunci akses pengguna IAM dan kunci akses pengguna akar) tidak dapat digunakan dengan akses API yang dilindungi MFA 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.
- Saat Anda mengizinkan orang lainAkun AWS mengakses sumber daya di akun Anda, keamanan sumber daya Anda tergantung pada konfigurasi akun tepercaya (akun lain, bukan akun 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 Anda mengizinkan anggota akun lain mengakses sumber daya AWS Anda yang memerlukan autentikasi multifaktor, 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 tepercaya.
- 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 lebih lanjut tentang pendelegasian lintas akun, lihat [Istilah 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 utama dalam kebijakan kepercayaan peran ke ID akun B. Kebijakan kepercayaan memberikan izin `AWS STS AssumeRole` tindakan tersebut. Anaya juga menambahkan ketentuan MFA ke kebijakan kepercayaan, seperti dalam contoh berikut.

```
{
  "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; memungkinkan pengguna yang mengasumsikan untuk melakukan tindakan Amazon DynamoDB apa pun pada tabel `Books` di akun A. kebijakan ini juga memungkinkan `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.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "TableActions",
      "Effect": "Allow",
      "Action": "dynamodb:*",
      "Resource": "arn:aws:dynamodb:*:ACCOUNT-A-ID: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 dia 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.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": ["sts:AssumeRole"],
    "Resource": ["arn:aws:iam::ACCOUNT-A-ID: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.

Saat Richard memanggil `AssumeRole`, AWS menentukan apakah ia memiliki kredensial yang valid, termasuk persyaratan untuk MFA. Jika demikian, Richard berhasil melaksanakan peran tersebut dan dapat melaksanakan tindakan DynamoDB di 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 operasi API sensitif hanya ketika pengguna diautentikasi menggunakan perangkat AWS MFA. Akun AWS

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 Sofia. Pengguna Anaya adalah administrator di akun A.

1. Anaya memastikan Sofia dikonfigurasi IT perangkat MFA AWS dan Sofia mengetahui ID perangkat tersebut. 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 Sofia 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.

```
{
  "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 Sofia harus menghentikan atau mengakhiri instans Amazon EC2, pengguna Sofia harus menghentikan atau mengakhiri instans Amazon, ia (atau aplikasi yang sedang berjalan) `GetSessionToken`. Operasi API ini meneruskan ID perangkat MFA dan TOTP saat ini yang diperoleh Sofia dari perangkatnya.

5. Pengguna Sofia (atau aplikasi yang digunakan Sofia) 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 berbeda Akun AWS, tetapi hanya jika pengguna tersebut terotentikasi 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.

```
{
  "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. Hapus MFA Amazon S3 tidak dapat diterapkan ke pengguna IAM, dan dikelola secara terpisah dari akses API yang dilindungi MFA. 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 mengizinkan 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": "AQoEXAMPLEH4aoAH0gNCAPyJxz4B1CFFxWNE1OPTgk5TthT  
+FvwqnKwRc0IfRrh3c/LTo6UDdyJw00vEVPvLXCrrrUtdnniCEXAMPLE/  
IvU1dYUg2RVAJBanLiHb4IgRmpRV3zrkuWJ0gQs8IZZaIv2BXIa2R40lgkBN9bkUDNCJiBeb/  
AX1zBBko7b15fjrBs2+cTQtpZ3CYWFXG8C5zqx37wn0E49mRl/+0tkIKG07fAE",  
    "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 [GetSessionToken](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Mengembalikan sebuah **Amazon.RuntimeAWSCredentials** instance yang berisi kredensial sementara yang valid untuk jangka waktu tertentu. Kredensial yang digunakan

untuk meminta kredensial sementara disimpulkan dari default shell saat ini. Untuk menentukan kredensial 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 'myprofile' 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...	SamPleTokenN.....

- Untuk detail API, lihat [GetSessionToken](#) di Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK for 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 [GetSessionToken](#) di 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

AWS SDK for .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>

        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);

// 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 [AssumeRole](#) di Referensi AWS SDK for .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 getopt 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 [AssumeRole](#) di Referensi AWS CLI Perintah.

C++

SDK for 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 [AssumeRole](#) di Referensi AWS SDK for 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": "AR0A3XFRBF535PLBIFPI4:s3-access-example",
    "Arn": "arn:aws:sts::123456789012:assumed-role/xaccounts3access/s3-access-example"
  },
  "Credentials": {
    "SecretAccessKey": "9drTJvcXLB89EXAMPLELB8923FB892xMFI",
    "SessionToken": "AQoXdzELDDY//////////
wEaoAK1wvxJY12r2IrDFT2IvAzTCn3zHoZ7YNtpiQLF0MqZye/
qwjzP2iEXAMPLEbw/m3hsj8VBTKPORGvr9jM5sgP+w9IZWZnU+LWhmg
+a5fDi2oTGUYcdg9uexQ4mtCHIHfi4citgqZTgco40Yqr4lIlo4V2b2Dyauk0eYFNebHtY1FVgAUj
+7Indz3LU0aTWk1WKIjHmMCIoTkyYp/k7kUG7moeEYKSitwQIi6Gjn+nyzM
+PtoA3685ixzv0R7i5rjQi0YE0lflloeie3bDiNHncmzosRM6SFiPzSvp6h/32xQuZsjcypmwsPSDtTPYcs0+YN/8B
IcrxSpnWEXAMPLEXSDFTAQAM6D19zR0tXoybnlrZIwML1Mi1Kcgo50ytwU=",
    "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 kredensial untuk Anda. Untuk informasi selengkapnya, lihat [Menggunakan peran IAM di AWS CLI](#) di Panduan Pengguna AWS CLI.

- Untuk detail API, lihat [AssumeRole](#) di Referensi AWS CLI Perintah.

Java

SDK for 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, rn: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 [AssumeRole](#) di 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 [AssumeRole](#) di Referensi AWS SDK for 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 [AssumeRole](#) di Referensi AWS SDK for JavaScript API.

PowerShell

Alat untuk PowerShell

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. Kredensial 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 kredensial 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 [AssumeRole](#) di Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK for 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 kredensial 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 [AssumeRole](#) di AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK for 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 [AssumeRole](#) di Referensi AWS SDK for 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 [AssumeRole](#) referensi AWS SDK for Rust API.

Swift

SDK for Swift

Note

Ini adalah dokumentasi prarilis untuk SDK dalam rilis pratinjau. Dokumentasi ini dapat berubah.

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
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 {
        throw error
    }
}
```

- Untuk detail API, lihat referensi [AssumeRole AWSSDK](#) untuk Swift API.

Temukan kredensi yang tidak digunakan AWS


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 AWS Management Console 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 AWS Management Console 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 AWS Management Console 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 kredensial \(AWS CLI\)](#).

Menemukan access key yang tidak digunakan

Anda dapat menggunakan AWS Management Console 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 AWS Management Console 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 AWS Management Console 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/A jika pengguna tidak memiliki kunci akses atau pengguna belum menggunakan kunci akses sejak IAM mulai melacak usia kunci akses pada 22 April 2015.

Untuk menemukan access key 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 kredensial \(AWS CLI\)](#).

Mendapatkan laporan kredensi untuk Anda Akun AWS

Anda dapat membuat dan mengunduh laporan kredensial yang 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 AWS Management Console, [AWSSDK](#) dan [Command Line Tools](#), atau IAM API.

Anda dapat menggunakan laporan kredensial untuk membantu upaya audit dan kepatuhan Anda. Anda dapat menggunakan laporan untuk mengaudit efek persyaratan siklus hidup kredensial, seperti kata sandi dan pembaruan 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 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 kredensial \(AWS CLI\)](#)
- [Mendapatkan laporan kredensial \(API AWS\)](#)

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 untuk Pengguna root akun AWS selalu `not_supported`.

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](#). AWSsitus web yang menangkap waktu masuk terakhir pengguna adalah AWS Management Console, Forum 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 kata sandi terakhir menggunakan informasi untuk mengidentifikasi kredensial yang tidak digunakan untuk penghapusan, seperti menghapus pengguna yang tidak masuk ke AWS dalam 90 hari terakhir, kami sarankan Anda menyesuaikan jendela evaluasi untuk menyertakan tanggal setelah 23 Mei 2018. Selain itu, jika pengguna menggunakan access key untuk mengakses AWS secara terprogram, Anda dapat merujuk ke informasi access key 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). Nilai untuk Akun AWS (root) selalunot_supported.

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) selalunot_supported.

mfa_active

Saat perangkat [otentikasi 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.

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).

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.

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.

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 yang terakhir digunakan tidak spesifik Wilayah, seperti Amazon S3

akses_key_1_last_used_service

Layanan AWS yang paling baru diakses dengan access key. 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.

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.

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, lihat [Memperbarui kunci akses](#).

access_key_2_last_rotated

Tanggal dan waktu, dalam [format tanggal-waktu ISO 8601](#), ketika 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).

akses_key_2_last_used_date

Tanggal dan waktu, dalam [format tanggal-waktu ISO 8601](#), saat access key kedua pengguna terakhir 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.

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. 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 yang terakhir digunakan tidak spesifik Wilayah, seperti Amazon S3

`access_key_2_last_used_service`

Layanan AWS yang paling baru diakses dengan access key kedua. 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. 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`


Saat pengguna memiliki sertifikat tanda tangan X.509 dan status sertifikat tersebut adalah `Active`, nilainya 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`

Saat pengguna memiliki sertifikat tanda tangan X.509 kedua dan status sertifikat tersebut adalah `Active`, nilainya adalah `TRUE`. Jika tidak, nilainya `FALSE`.

 **Note**

Pengguna dapat memiliki hingga dua sertifikat tanda tangan X.509, untuk membuat rotasi sertifikat lebih mudah.

`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).

Mendapatkan laporan kredensial (konsol)

Anda dapat menggunakan AWS Management Console untuk mengunduh laporan kredensial sebagai file nilai yang dipisahkan koma (CSV).

Untuk mengunduh laporan kredensial (konsol)

1. Masuk ke AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Di panel navigasi, pilih Laporan kredensial.
3. Pilih Unduh Laporan.

Mendapatkan laporan kredensial (AWS CLI)

Untuk mengunduh laporan kredensial (AWS CLI)

1. Buat laporan kredensial. AWS menyimpan laporan tunggal. 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 kredensial (API AWS)

Untuk mengunduh laporan kredensial (API AWS)

1. Buat laporan kredensialnya. AWS menyimpan laporan tunggal. Jika ada laporan, membuat laporan kredensial akan menimpa laporan sebelumnya. [`GenerateCredentialReport`](#)
2. Melihat laporan terakhir yang dihasilkan: [`GetCredentialReport`](#)

Menggunakan IAM dengan CodeCommit: Kredensial 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 kredensial:

- Kredensial 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.
- [AWSkunci akses](#), yang dapat Anda gunakan dengan pembantu kredensial 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 lainAkun 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 kredensial 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 kredensial 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 [Membuat pengguna IAM di Akun AWS](#).
- Untuk membuat dan menggunakan kredensial 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 [Membuat 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 kredenial 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 dan federasi identitas](#) dan [Menyediakan 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 kredensial AWS CLI tidak kompatibel dengan sistem pembantu kredensial lainnya, seperti Keychain Access atau Windows Credential Management. 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

Menggunakan 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 AWS Management Console, atau secara terprogram. Untuk mengakses Amazon Keyspaces secara terprogram dengan kredensial khusus layanan, Anda dapat menggunakan atau driver Cassandra sumber terbuka. `cqlsh` Kredensial khusus layanan mencakup nama pengguna dan kata sandi seperti yang digunakan Cassandra untuk otentikasi dan manajemen akses. Anda dapat memiliki maksimal dua set kredensial 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 lebih lanjut, lihat [Mengakses Amazon Keyspaces dalam 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 lebih lanjut, lihat [Mengakses Amazon Keyspaces menggunakan konsol dalam Panduan Pengembang Amazon Keyspaces](#) (untuk Apache Cassandra).

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 Amazon Keyspaces (AWS Management Console untuk Apache Cassandra) untuk pengguna IAM Anda.

Untuk menghasilkan kredensial khusus layanan Amazon Keyspaces (konsol)

1. Masuk ke AWS Management Console 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 Amazon Keyspaces (AWS CLI untuk Apache Cassandra) untuk pengguna IAM Anda.

Untuk menghasilkan kredensi khusus layanan Amazon Keyspaces () AWS CLI

- Gunakan perintah berikut:
 - [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 kredensial khusus layanan Amazon Keyspaces (API) AWS

- Selesaikan operasi berikut:
 - [CreateServiceSpecificCredential](#)

Mengelola sertifikat server di IAM

Untuk mengaktifkan koneksi HTTPS ke situs web atau aplikasi Anda di AWS, Anda memerlukan sertifikat server SSL/TLS. Untuk sertifikat di Wilayah yang didukung oleh AWS Certificate Manager (ACM), kami menyarankan agar Anda menyediakan, mengelola, dan menerapkan sertifikat server Anda. Di Wilayah yang tidak didukung, Anda harus menggunakan IAM sebagai manajer sertifikat. Untuk mempelajari Regions yang didukung ACM, lihat [AWS Certificate Manager titik akhir dan kuota](#) di bagian Referensi Umum AWS.

ACM adalah alat pilihan untuk menyediakan, mengelola, dan menerapkan sertifikat server Anda. Dengan ACM Anda dapat meminta sertifikat atau menerapkan ACM yang ada atau sertifikat eksternal keAWS 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 tentang cara menggunakan ACM, lihat [PanduanAWS Certificate Manager Pengguna](#). Untuk informasi selengkapnya tentang meminta sertifikat ACM, lihat [Meminta Sertifikat Publik](#) atau [Minta Sertifikat Privat](#) di PanduanAWS 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 menerapkan sertifikat server di semua Wilayah, tetapi Anda harus mendapatkan sertifikat dari penyedia eksternal untuk menggunakan digunakan denganAWS. 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.

Daftar Isi

- [Mengunggah sertifikat server \(API AWS\)](#)
- [Mengambil sertifikat server \(API AWS\)](#)
- [Daftar sertifikat server \(API AWS\)](#)
- [Penandaan dan Penghapusan Tanda Sertifikat Server \(API AWS\)](#)
- [Mengganti nama sertifikat server atau memperbarui jalurnya \(API AWS\)](#)
- [Menghapus sertifikat server \(API AWS\)](#)
- [Pemecahan Masalah](#)

Mengunggah sertifikat server (API AWS)

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 [Pemecahan Masalah](#).
- Sertifikat, kunci pribadi, dan rantai sertifikat harus semuanya dikode PEM. Untuk membantu mengonversi item ini ke format PEM, lihat [Pemecahan Masalah](#).

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:

- Sertifikat yang diencode PEM disimpan dalam file dengan nama `Certificate.pem`.
- Rantai sertifikat yang diencode PEM disimpan dalam file dengan nama `CertificateChain.pem`.
- Sertifikat yang diencode PEM, kunci pribadi tidak dienkripsi, disimpan dalam file dengan nama `PrivateKey.pem`.
- (Opsional) Anda ingin menandai sertifikat server dengan pasangan nilai—kunci. 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 unggahan Anda.

Jika Anda ingin menandai sertifikat, ganti pasangan nilai-kunci dengan nilai Anda sendiri *ExampleKeyExampleValue*. 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 menggunakan digunakan dengan Amazon CloudFront, Anda harus menentukan alur menggunakan--path pilihan. 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](#).

Mengambil sertifikat server (API AWS)

Untuk menggunakan API IAM untuk mengunggah sertifikat, kirim permintaan [GetServerCertificate](#). Contoh berikut ini menunjukkan cara melakukan ini dengan AWS CLI. Ganti *ExampleCertificate* dengan nama sertifikat untuk diambil.

```
aws iam get-server-certificate --server-certificate-name ExampleCertificate
```

Ketika perintah sebelumnya berhasil, perintah mengembalikan sertifikat, rantai sertifikat (jika salah satu diunggah), dan metadata tentang sertifikat tersebut.

Note

Anda tidak dapat mengunduh atau mengambil kunci pribadi dari IAM setelah Anda mengunggahnya.

Untuk menggunakan AWS Tools for Windows PowerShell untuk mengambil sertifikat, gunakan [Get-IAMServerCertificate](#).

Daftar sertifikat server (API AWS)

Untuk menggunakan API IAM untuk mengunggah sertifikat, kirim permintaan [ListServerCertificates](#). Contoh berikut ini menunjukkan cara melakukan ini dengan AWS CLI.


```
aws iam list-server-certificates
```

Jika perintah sebelumnya berhasil, perintah akan mengembalikan daftar yang berisi metadata setiap sertifikat.

Untuk menggunakan daftar sertifikat server yang Anda unggah, gunakan [Get-IAMServerCertificates](#). AWS Tools for Windows PowerShell

Penandaan dan Penghapusan Tanda Sertifikat Server (API AWS)

Anda dapat melampirkan tanda ke sumber daya IAM untuk mengatur dan mengontrol akses ke mereka. Untuk menggunakan API IAM untuk menandai sertifikat server yang ada, kirim [TagServerCertificate](#) permintaan. Contoh berikut ini menunjukkan cara melakukan ini dengan AWS CLI.

```
aws iam tag-server-certificate --server-certificate-name ExampleCertificate
                                --tags '{"Key": "ExampleKey", "Value":
                                "ExampleValue"}'
```

Ketika perintah sebelumnya berhasil, tidak ada output yang dihasilkan.

Untuk menggunakan API IAM untuk mengunggah sertifikat server, kirim [UntagServerCertificate](#) permintaan. Contoh berikut ini menunjukkan cara melakukan ini dengan AWS CLI.

```
aws iam untag-server-certificate --server-certificate-name ExampleCertificate
                                --tag-keys ExampleKeyName
```

Ketika perintah sebelumnya berhasil, tidak ada output yang dihasilkan.

Mengganti nama sertifikat server atau memperbarui jalurnya (API AWS)

Untuk menggunakan API IAM untuk mengganti nama sertifikat server atau memperbarui jalurnya, kirim permintaan [UpdateServerCertificate](#). 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/
```

Ketika perintah sebelumnya berhasil, itu tidak mengembalikan keluaran apa pun.

Untuk menggunakan AWS Tools for Windows PowerShell untuk mengganti nama sertifikat server atau memperbarui jalurnya, gunakan [Update-IAMServerCertificate](#).

Menghapus sertifikat server (API AWS)

Untuk menggunakan API IAM untuk menghapus sertifikat server, kirim permintaan [DeleteServerCertificate](#). Contoh berikut ini menunjukkan cara melakukan ini dengan AWS CLI.

Untuk menggunakan perintah contoh berikut, ganti *ExampleCertificate* dengan nama sertifikat untuk dihapus.

```
aws iam delete-server-certificate --server-certificate-name ExampleCertificate
```

Ketika perintah sebelumnya berhasil, itu tidak mengembalikan keluaran apa pun.

Untuk menggunakan AWS Tools for Windows PowerShell untuk menghapus sertifikat server, gunakan [Hapus-IAMServerCertificate](#).

Pemecahan Masalah

Sebelum Anda dapat mengunggah sertifikat ke IAM, Anda harus memastikan bahwa sertifikat, kunci pribadi, dan rantai sertifikat semuanya dikode PEM. Anda juga harus memastikan bahwa kunci pribadi tidak terenkripsi. Lihat contoh-contoh berikut.

Example Contoh sertifikat yang diencode PEM

```
-----BEGIN CERTIFICATE-----
Base64-encoded certificate
-----END CERTIFICATE-----
```

Example Contoh kunci pribadi yang diencode PEM, tidak terenkripsi

```
-----BEGIN RSA PRIVATE KEY-----
Base64-encoded private key
-----END RSA PRIVATE KEY-----
```

Example Contoh rantai sertifikat yang diencode PEM

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 bukan dalam format yang tepat untuk mengunggah ke IAM, Anda dapat menggunakan [OpenSSL](#) untuk mengonversinya ke format yang tepat.

Untuk mengonversikan sertifikat atau rantai sertifikat dari DER ke PEM

Gunakan [perintah OpenSSL x509](#), seperti pada contoh berikut. Di perintah contoh berikut, ganti *Certificate.der* dengan nama file yang berisi sertifikat yang diencode DER Anda. Ganti *Certificate.pem* dengan nama pilihan file output untuk memuat sertifikat yang diencode PEM.

```
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. Di perintah contoh berikut, ganti *PrivateKey.der* dengan nama file yang berisi kunci pribadi yang diencode DER Anda. Ganti *PrivateKey.pem* dengan nama pilihan file output untuk memuat kunci pribadi yang diencode PEM.

```
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 memuat kunci pribadi tidak terenkripsi yang dienkode PEM.

```
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 pilihan file output untuk memuat bundel sertifikat yang dienkode PEM.

```
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 pilihan file output untuk memuat bundel sertifikat yang dienkode PEM.

```
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

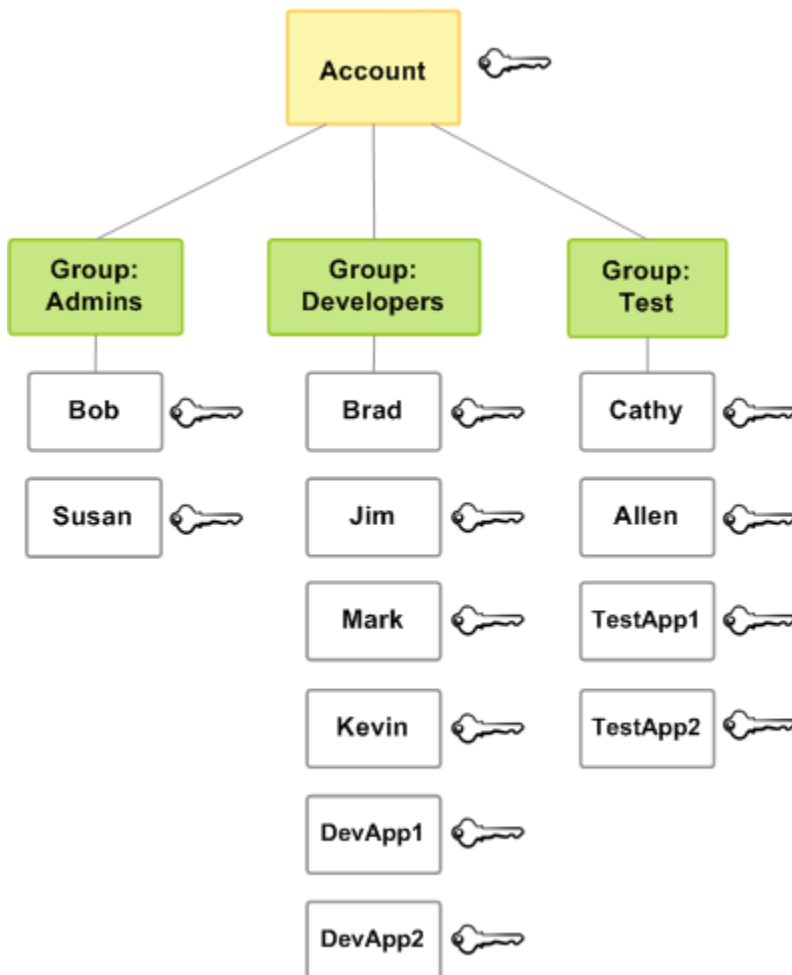
pengguna tersebut. Misalnya, Anda dapat meminta grup pengguna untuk menghubungi Admin dan memberikan izin administrator grup pengguna yang khas. Setiap pengguna dalam grup pengguna tersebut secara otomatis memiliki izin grup Admin. Jika pengguna baru bergabung dengan organisasi Anda dan memerlukan hak administrator, Anda dapat menetapkan izin yang sesuai dengan menambahkan pengguna ke grup pengguna Admin. Jika seseorang berganti pekerjaan dalam organisasi Anda, alih-alih mengedit izin pengguna tersebut, Anda dapat menghapusnya dari grup pengguna lama dan menambahkannya ke grup pengguna baru yang sesuai.

Anda dapat melampirkan kebijakan berbasis identitas ke grup pengguna sehingga semua pengguna di grup pengguna menerima izin kebijakan. Anda tidak dapat mengidentifikasi grup pengguna sebagai `Principal` dalam kebijakan (seperti kebijakan berbasis sumber daya) karena grup terkait dengan izin, bukan otentikasi, dan prinsipal adalah entitas IAM yang diautentikasi. Untuk informasi lebih lanjut tentang jenis kebijakan, lihat [Kebijakan berbasis identitas dan kebijakan berbasis sumber daya](#).

Berikut adalah beberapa karakteristik penting dari grup pengguna:

- Sebuah grup pengguna dapat berisi banyak pengguna, dan pengguna dapat menjadi bagian dari beberapa grup pengguna.
- Grup pengguna tidak dapat dibuat bertingkat; grup hanya dapat berisi pengguna, bukan grup pengguna lain.
- 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 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 pengguna `Admins` bagi pengguna untuk membuat dan mengelola pengguna lain seiring pertumbuhan perusahaan. Parameter grup pengguna `Admins` membuat grup pengguna `Developers` dan grup pengguna `Test`. Masing-masing kelompok ini terdiri atas pengguna (manusia dan aplikasi) yang berinteraksi dengan AWS (Jim, Brad, DevApp 1, dan sebagainya). 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 pengguna.



Membuat grup pengguna IAM

Note

Sebagai [praktik terbaik](#), kami menyarankan Anda meminta pengguna manusia untuk menggunakan federasi dengan penyedia identitas untuk mengakses AWS menggunakan kredensial sementara. Jika Anda mengikuti praktik terbaik, Anda tidak mengelola pengguna dan grup IAM. Sebagai gantinya, 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 kredensial 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 IAM Identity Center](#) di


PanduanAWS IAM Identity Center Pengguna untuk informasi tentang membuat pengguna dan grup di IAM Identity Center.

Untuk mengatur grup pengguna, Anda perlu membuat grup. Kemudian berikan izin grup berdasarkan jenis pekerjaan yang Anda harapkan dilakukan pengguna dalam grup. Terakhir, tambahkan pengguna ke grup.

Untuk informasi tentang izin yang Anda butuhkan untuk membuat grup pengguna, lihat [Izin diperlukan untuk mengakses sumber daya IAM](#).

Untuk membuat grup pengguna IAM dan melampirkan ((

1. Masuk ke AWS Management Console 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 diAWS 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 (@), garis bawah (_), dan tanda hubung (-). Nama harus unik dalam akun. Grup tidak dibedakan berdasarkan huruf besar-kecil. 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 Create group (Buat grup).

Untuk membuat grup pengguna IAM (AWS CLIatauAWS API)

Gunakan salah satu langkah berikut:

- AWS CLI: [aws iam create-group](#)
- AWS API: [CreateGroup](#)

Mengelola grup pengguna IAM

Amazon Web Services menawarkan beberapa alat untuk mengelola grup pengguna IAM. Untuk informasi tentang izin yang Anda butuhkan untuk menambahkan atau menghapus pengguna di grup pengguna, lihat [Izin diperlukan untuk mengakses sumber daya IAM](#).

Topik

- [Membuat daftar grup pengguna IAM](#)
- [Menambahkan dan menghapus pengguna di grup pengguna IAM](#)
- [Melampirkan kebijakan ke grup pengguna IAM](#)
- [Mengganti nama sebuah grup pengguna IAM](#)
- [Menghapus grup pengguna IAM](#)

Membuat daftar grup pengguna IAM

Anda dapat mencantumkan semua grup pengguna di akun Anda, mencantumkan pengguna di grup pengguna, dan mencantumkan grup pengguna tempat pengguna berada. Apabila Anda menggunakan AWS CLI atau API AWS, Anda dapat mencantumkan semua grup pengguna dengan prefiks jalur tertentu.

Untuk mencantumkan semua grup pengguna di akun Anda

Lakukan langkah-langkah berikut:

- [AWS Management Console](#): Di panel navigasi, pilih Grup pengguna.
- AWS CLI: [aws iam list-groups](#)
- AWSAPI: [ListGroup](#)

Untuk mencantumkan pengguna dalam grup pengguna tertentu

Lakukan langkah-langkah berikut:

- [AWS Management Console](#): Di panel navigasi, pilih Grup pengguna, pilih nama grup, lalu pilih tab Pengguna.

- AWS CLI: [aws iam get-group](#)
- AWSAPI: [GetGroup](#)

Untuk mencantumkan semua grup pengguna tempat pengguna berada

Lakukan langkah-langkah berikut:

- [AWS Management Console](#): Di panel navigasi, pilih Pengguna, pilih nama pengguna, lalu pilih tab Grup.
- AWS CLI: [AWS list-groups-for-user](#)
- AWS API: [ListGroupForUser](#)

Menambahkan dan menghapus pengguna di grup pengguna IAM

Gunakan grup pengguna untuk menerapkan kebijakan izin yang sama ke beberapa pengguna sekaligus. Anda kemudian dapat menambahkan pengguna ke atau menghapus pengguna dari grup pengguna IAM. Hal ini berguna saat orang-orang memasuki dan meninggalkan organisasi Anda.

Lihat akses kebijakan

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 tentang melihat informasi yang terakhir diakses, lihat [Menyaring izin di AWS menggunakan informasi yang terakhir diakses](#).

Menambahkan atau menghapus pengguna dalam grup pengguna (konsol)

Anda dapat menggunakan AWS Management Console untuk menambah atau menghapus pengguna dari grup pengguna.

Untuk menambahkan pengguna ke grup pengguna IAM (konsol)

1. Masuk ke AWS Management Console 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.

Untuk menghapus pengguna dari grup IAM (konsol)

1. Masuk ke AWS Management Console 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.

Menambahkan atau menghapus pengguna dalam grup pengguna (AWS CLI)

Anda dapat menggunakan AWS CLI untuk menambah atau menghapus pengguna dari grup pengguna.

Untuk menambahkan pengguna ke grup pengguna IAM (AWS CLI)

- Gunakan perintah berikut:
 - [AWS add-user-to-group](#)

Untuk menghapus pengguna dari grup pengguna IAM (AWS CLI)

- Gunakan perintah berikut:
 - [AWS remove-user-from-group](#)

Menambahkan atau menghapus pengguna dalam grup pengguna (API AWS)

Anda dapat menggunakan API AWS untuk menambah atau menghapus pengguna dalam grup pengguna.

Untuk menambahkan pengguna ke grup IAM (AWSAPI)

- Selesaikan operasi berikut:
 - [AddUserToGroup](#)

Untuk menghapus pengguna dari grup pengguna IMAWS

- Selesaikan 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 [Membuat kebijakan IAM](#) .

Untuk informasi lebih lanjut tentang izin dan kebijakan, lihat [Manajemen akses untuk sumber daya AWS](#).

Untuk melampirkan kebijakan ke grup pengguna (konsol)

1. Masuk ke AWS Management Console 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 pengguna IAM Anda dan pilih Lampirkan kebijakan.

Untuk melampirkan kebijakan ke grup pengguna (API AWS CLI atau AWS)

Lakukan salah satu dari langkah berikut:

- AWS CLI: [aws iam attach-group-policy](#)
- AWS API: [AttachGroupPolicy](#)

Mengganti nama sebuah 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 selengkapnya tentang ID 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 suatu grup pengguna 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 di dalam kebijakan.

Untuk mengganti nama grup pengguna IAM

Lakukan langkah-langkah berikut:

- [AWS Management Console](#): Di panel navigasi, pilih Grup pengguna lalu pilih nama grup. Pilih Edit. Ketik nama grup pengguna baru lalu pilih Simpan perubahan.
- AWS CLI: [aws iam update-group](#)
- AWS API: [UpdateGroup](#)

Menghapus grup pengguna IAM

Ketika Anda menghapus grup pengguna di AWS Management Console, konsol secara otomatis menghapus semua anggota grup, melepaskan semua kebijakan terkelola yang terlampir, dan menghapus semua kebijakan inline. Namun, karena IAM tidak secara otomatis menghapus kebijakan yang merujuk ke grup pengguna sebagai sumber daya, Anda harus berhati-hati saat menghapus grup pengguna. Sebelum menghapus grup pengguna, Anda harus memeriksa semua kebijakan Anda secara manual untuk menemukan kebijakan yang menyebutkan nama grup tersebut. 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 Penguji.

Untuk menemukan kebijakan yang merujuk suatu grup pengguna 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 dihapus.
4. Pilih Izin, dan kemudian pilih Ringkasan.
5. Pilih IAM dari daftar layanan, jika ada.
6. Cari nama grup pengguna Anda di kolom Sumber Daya.
7. Pilih Hapus untuk menghapus kebijakan.
8. Ketik nama kebijakan untuk mengonfirmasi penghapusan kebijakan dan pilih Hapus.

Sebaliknya, saat Anda menggunakan AWS CLI, Alat untuk Windows PowerShell, atau AWS API untuk menghapus grup pengguna, Anda harus terlebih dahulu menghapus pengguna dalam grup. Kemudian hapus kebijakan inline apa pun yang disematkan dalam grup pengguna. Berikutnya, lepaskan kebijakan terkelola yang melekat pada grup. Hanya dengan begitu Anda dapat menghapus grup pengguna itu sendiri.

Menghapus grup pengguna IAM (konsol)

Anda dapat menghapus grup pengguna IAM dari AWS Management Console.

Untuk menghapus grup pengguna IAM (konsol)

1. Masuk ke AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Di panel navigasi, pilih Grup pengguna.
3. Di dalam daftar grup pengguna, pilih kotak centang di samping nama grup pengguna yang akan dihapus. Anda dapat menggunakan kotak pencarian untuk memfilter daftar grup pengguna menurut jenis, izin, dan nama grup pengguna.
4. Pilih Hapus.
5. Dalam kotak konfirmasi, jika Anda ingin menghapus satu grup pengguna, ketik nama grup pengguna dan pilih Hapus. Jika Anda ingin menghapus beberapa kelompok pengguna, ketik jumlah grup pengguna yang ingin dihapus diikuti oleh **user groups** dan pilih Hapus. Sebagai contoh, jika Anda menghapus tiga grup pengguna, ketik **3 user groups**.

Menghapus grup pengguna IAM (AWS CLI)

Anda dapat menghapus grup pengguna IAM dari AWS CLI.

Untuk menghapus grup pengguna IAM (AWS CLI)

1. Menghapus semua pengguna dari grup pengguna.
 - [aws iam get-grup](#) (untuk mendapatkan daftar pengguna dalam grup pengguna), dan [AWSremove-user-from-group](#) (untuk menghapus pengguna dari grup pengguna)
2. Menghapus semua kebijakan inline yang disertakan dalam grup pengguna.
 - [AWSlist-group-policies](#) (untuk mendapatkan daftar kebijakan inline grup pengguna), dan [AWSdelete-group-policy](#) (untuk menghapus kebijakan inline grup pengguna)
3. Melepaskan semua kebijakan terkelola yang terlampir pada grup pengguna.
 - [AWSlist-attached-group-policies](#) (untuk mendapatkan daftar kebijakan terkelola yang dilampirkan ke grup pengguna), dan [AWSdetach-group-policy](#) (untuk melepaskan kebijakan terkelola dari grup pengguna)
4. Menghapus grup pengguna.

- [aws iam menghapus-grup](#)

Menghapus grup pengguna IAM (AWSAPI)

Anda dapat menggunakan AWSAPI untuk menghapus grup pengguna IAM.

Untuk menghapus grup pengguna IAM (AWSAPI)

1. Menghapus semua pengguna dari grup pengguna.
 - [GetGroup](#) (untuk mendapatkan daftar pengguna dalam grup pengguna) dan [RemoveUserFromGroup](#) (untuk menghapus pengguna dari grup pengguna)
2. Menghapus semua kebijakan inline yang disertakan dalam grup pengguna.
 - [ListGroupPolicies](#) (untuk mendapatkan daftar kebijakan inline grup pengguna) dan [DeleteGroupPolicy](#) (untuk menghapus kebijakan inline grup pengguna)
3. Melepaskan semua kebijakan terkelola yang terlampir pada grup pengguna.
 - [ListAttachedGroupPolicies](#) (untuk mendapatkan daftar kebijakan terkelola yang dilampirkan ke grup pengguna) dan [DetachGroupPolicy](#) (untuk melepaskan kebijakan terkelola dari grup pengguna)
4. Menghapus grup pengguna.
 - [DeleteGroup](#)

Peran IAM

Sebuah role IAM adalah identitas IAM yang dapat Anda buat di akun Anda yang memiliki izin spesifik. 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 menyimpan AWS kunci di dalam aplikasi (di mana kunci tersebut sulit 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.

Note

Saat 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 penggunaannya.

Peran terkait layanan adalah jenis peran layanan yang ditautkan ke. Layanan AWS Layanan dapat menggunakan peran untuk melakukan tindakan atas nama Anda. Peran terkait layanan 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 terlebih dahulu menghapus sumber daya terkait mereka. 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 peran yang terkait dengan layanan, lihat [AWS layanan yang bekerja dengan IAM](#) dan cari layanan yang memiliki Ya di kolom Peran Terkait-Layanan Pilih Ya dengan tautan untuk melihat dokumentasi peran terkait layanan untuk layanan tersebut.

Topik

- [Istilah dan konsep peran](#)
- [Skenario umum untuk peran: Pengguna, aplikasi, dan layanan](#)
- [Menggunakan peran terkait layanan](#)
- [Membuat Peran IAM](#)

- [Menggunakan peran IAM](#)
- [Mengelola peran IAM](#)

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 dapat menjadi dapat diasumsikan 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, kredensial keamanan sementara untuk sesi peran Anda akan diberikan.

Peran dapat digunakan sebagai berikut:

- Pengguna IAM dalam peran yang Akun AWS sama
- Pengguna IAM dalam peran yang berbeda Akun AWS dari
- Layanan web yang ditawarkan oleh AWS Amazon Elastic Compute Cloud (Amazon EC2)
- Pengguna eksternal yang diautentikasi oleh layanan penyedia identitas (IdP) eksternal yang kompatibel dengan SAML 2.0 atau OpenID Connect, atau broker identitas yang dirancang khusus.

AWS peran layanan

Peran layanan adalah [peran IAM](#) yang diambil oleh layanan untuk melakukan tindakan atas nama Anda. Administrator IAM dapat membuat, mengubah, dan menghapus peran layanan dari dalam IAM. Untuk informasi selengkapnya, lihat [Membuat peran untuk mendelegasikan izin ke Layanan AWS](#) dalam Panduan pengguna IAM.

AWS peran layanan untuk instans EC2

Jenis peran layanan khusus yang dapat dijalankan aplikasi yang dijalankan di instans Amazon EC2 dapat mengambil tindakan di akun Anda. Peran ini ditetapkan ke instans EC2 saat peran diluncurkan. Aplikasi yang berjalan pada instans tersebut dapat mengambil kredensial keamanan sementara dan melakukan tindakan yang dimungkinkan oleh peran tersebut. Untuk detail tentang

penggunaan peran layanan untuk instans EC2, lihat [Menggunakan peran IAM untuk memberikan izin pada aplikasi yang berjalan di instans Amazon EC2](#).

AWS peran terkait layanan

Peran terkait layanan adalah jenis peran layanan yang ditautkan ke. Layanan AWS Layanan dapat menggunakan peran untuk melakukan tindakan atas nama Anda. Peran terkait layanan 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 akun AWS saya](#).

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 Pilih Ya dengan tautan untuk melihat dokumentasi peran terkait layanan untuk layanan tersebut. Untuk informasi selengkapnya, lihat [Menggunakan peran terkait layanan](#).

Rantai peran

Role chaining adalah ketika Anda menggunakan peran untuk mengambil peran kedua melalui AWS CLI atau API. Misalnya, RoleA memiliki izin untuk berasumsiRoleB. Anda dapat mengaktifkan User1 untuk berasumsi RoleA dengan menggunakan kredensi pengguna jangka panjang mereka dalam operasi API. AssumeRole Ini mengembalikan kredensi RoleA jangka pendek. Dengan rantai peran, Anda dapat menggunakan RoleA kredensi jangka pendek untuk mengaktifkan User1 berasumsi. 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 [Melewati tag sesi di AWS STS](#).

Role chaining membatasi sesi peran Anda AWS CLI atau AWS API hingga maksimal satu jam. 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.

AWS tidak memperlakukan penggunaan peran untuk [memberikan izin ke aplikasi yang berjalan pada instans EC2](#) sebagai rantai peran.

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 terpercaya). Akun terpercaya 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 [menciptakan peran IAM](#) dalam akun kepercayaan yang memiliki dua [kebijakan](#) yang melekat. Kebijakan izin memberikan izin yang diperlukan pengguna untuk melaksanakan tugas yang diinginkan pada sumber daya. Kebijakan kepercayaan menentukan anggota akun terpercaya mana yang diizinkan untuk menjalankan peran tersebut.

Saat membuat kebijakan kepercayaan, Anda tidak dapat menentukan wildcard (*) sebagai bagian dari dan 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 terpercaya 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.

Federasi

Penciptaan hubungan kepercayaan antara penyedia identitas eksternal dan AWS. Pengguna dapat masuk ke penyedia OIDC, seperti Login with Amazon, Facebook, Google, atau IDP apa

pun yang kompatibel dengan OpenID Connect (OIDC). Pengguna juga dapat masuk ke sistem identitas perusahaan yang kompatibel dengan Security Assertion Markup Language (SAML) 2.0, seperti Microsoft Active Directory Federation Services. Saat Anda menggunakan OIDC dan SAMP 2.0 untuk mengonfigurasi hubungan kepercayaan antara penyedia identitas eksternal ini dan AWS, pengguna ditetapkan ke peran IAM. Pengguna juga menerima kredensi sementara yang memungkinkan pengguna mengakses sumber daya Anda AWS .

Pengguna federasi

Alih-alih membuat pengguna IAM, Anda dapat menggunakan identitas yang ada dari AWS Directory Service, direktori pengguna perusahaan Anda, atau penyedia OIDC. Ini dikenal sebagai pengguna federasi. AWS memberikan peran kepada pengguna federasi ketika akses diminta melalui penyedia [identitas](#). Untuk informasi selengkapnya tentang pengguna gabungan, lihat [Pengguna gabungan dan peran](#).

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.

Kebijakan izin

Dokumen izin di format [JSON](#) yang dapat Anda gunakan untuk menentukan tindakan dan sumber daya apa yang dapat digunakan oleh peran tersebut. Dokumen ini ditulis sesuai dengan aturan [Bahasa kebijakan IAM](#).

Batas izin

Fitur lanjutan di mana Anda menggunakan kebijakan untuk membatasi izin maksimum yang dapat diberikan oleh kebijakan berbasis identitas ke suatu peran. Anda tidak dapat menerapkan batas izin untuk peran yang terkait layanan. Untuk informasi selengkapnya, lihat [Batas izin untuk entitas IAM](#).

Kepala Sekolah

Entitas AWS yang dapat melakukan tindakan dan mengakses sumber daya. Prinsipal dapat berupa Pengguna root akun AWS, pengguna IAM, atau peran. Anda dapat memberikan izin untuk mengakses sumber daya dengan salah satu dari dua cara:

- Anda dapat melampirkan kebijakan izin kepada pengguna (secara langsung atau tidak langsung melalui grup) atau ke peran.

- Untuk layanan yang mendukung [kebijakan berbasis sumber daya](#), Anda dapat mengidentifikasi prinsipal dalam elemen `Principal` kebijakan yang terlampir pada sumber daya.

Jika Anda merujuk Akun AWS sebagai prinsipal, itu umumnya berarti setiap prinsipal yang didefinisikan dalam akun itu.

Note

Anda tidak dapat menggunakan wildcard (*) untuk mencocokkan bagian dari nama utama atau ARN dalam kebijakan kepercayaan peran. Untuk detailnya, lihat [AWS Elemen kebijakan JSON: Principal](#).

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 layanan AWS memungkinkan Anda melampirkan kebijakan secara langsung ke sumber daya (alih-alih menggunakan peran sebagai proksi). Ini disebut kebijakan berbasis sumber daya, dan Anda dapat menggunakannya untuk memberikan prinsipal di akses lain ke sumber daya. Akun AWS Beberapa sumber daya ini termasuk bucket Amazon Simple Storage Service (S3), vault S3 Glacier, topik Amazon Simple Notification Service (SNS) dan antrean 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](#).

Skenario umum untuk peran: Pengguna, aplikasi, dan layanan

Seperti AWS fitur kebanyakan fitur, Anda biasanya memiliki dua cara untuk menggunakan peran: secara interaktif di konsol IAM, atau secara terprogram dengan AWS CLI, Tools for Windows 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) bisa mengasumsikan peran dengan meminta kredensi keamanan sementara untuk peran yang akan membuat

permintaan terprogram keAWS. 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 di dalam akun Anda sendiri atau akun lainnyaAkun 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 yang masuk menggunakan kredensial dari penyedia identitas (IdP). Kemudian AWS memberikan kredensial sementara kepada IdP tepercaya untuk meneruskan kepada pengguna agar disertakan dalam permintaan sumber daya 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 satuAkun AWS yang Anda miliki untuk mengakses sumber daya di akun lain yang Anda miliki](#)
- [Menyediakan akses ke nonAWS beban kerja](#)
- [Menyediakan akses ke pengguna IAM yangAkun AWS dimiliki oleh pihak ketiga](#)
- [Menyediakan akses untuk layanan yang ditawarkan olehAWS keAWS sumber daya](#)
- [Menyediakan akses untuk pengguna yang diautentikasi secara eksternal \(federasi identitas\)](#)

Menyediakan akses ke pengguna IAM di pengguna lain Akun AWS yang Anda miliki

Anda dapat memberikan izin kepada pengguna IAM 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 [Menyediakan Akun AWS akses ke 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 AWS Management Console atau menganggap 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 [Mengonfigurasi akses API yang dilindungi 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 terhadap lingkungan yang sensitif, terutama jika Anda menggabungkannya dengan [audit](#) untuk membantu memastikan bahwa peran tersebut hanya digunakan saat diperlukan.

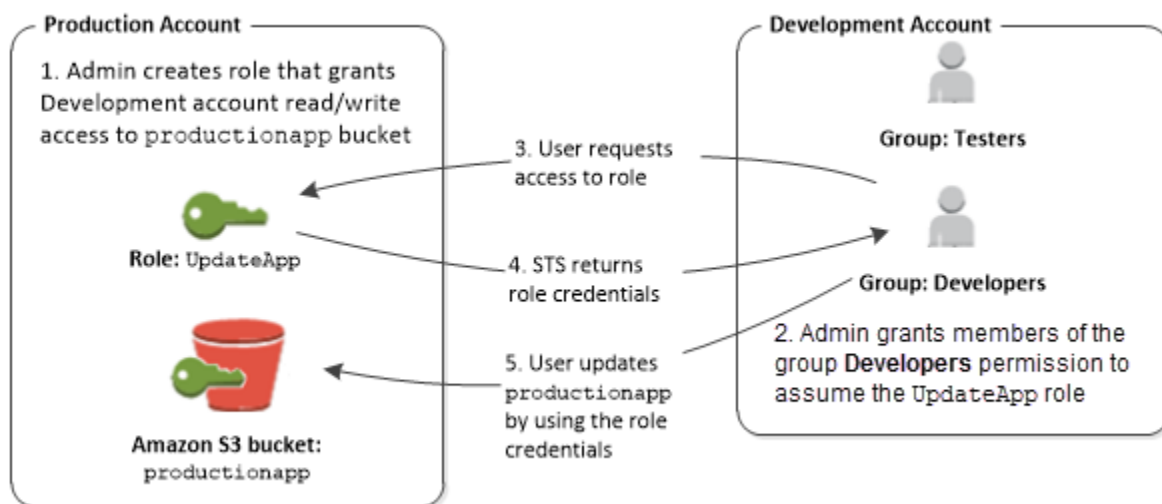
Saat Anda membuat peran untuk tujuan ini, Anda menetapkan akun berdasarkan ID yang penggunaannya memerlukan akses dalam elemen `Principal` kebijakan kepercayaan peran. Kemudian, Anda dapat memberikan izin kepada pengguna tertentu dalam akun lain untuk beralih ke peran tersebut. Untuk mempelajari apakah 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 beberapa 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 menetapkan kebijakan izin untuk peran yang menentukan izin baca dan tulis ke bucket Amazon S3 diberi nama `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 konsol AWS) atau Amazon Resource Name (ARN) (untuk akses AWS CLI atau API AWS). 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 [Mengonfigurasi akses API yang dilindungi MFA](#).

2. Dalam akun pengembangan, administrator memberi anggota grup Pengembang izin untuk beralih ke peran. Hal ini dilakukan dengan memberikan izin kepada grup Developer untuk memanggil AWS Security Token Service (AWS STS) API `AssumeRole` untuk peran `UpdateApp`. 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:
 - Konsol AWS: Pengguna memilih nama akun di bilah navigasi dan memilih Beralih Peran. Pengguna menentukan ID akun (atau alias) dan nama peran. Atau, pengguna dapat mengklik tautan yang dikirim melalui email oleh administrator. Tautan tersebut membawa pengguna ke halaman Beralih Peran dengan detail yang sudah diisi.
 - API AWS/AWS CLI: Pengguna dalam grup Developer dari akun pengembangan memanggil fungsi `AssumeRole` untuk mendapatkan kredensial bagi peran `UpdateApp`. 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:
 - Konsol AWS: AWS STS memverifikasi permintaan dengan kebijakan kepercayaan peran untuk memastikan bahwa permintaan tersebut berasal dari entitas tepercaya (yaitu: akun pengembangan). Setelah verifikasi, AWS STS mengembalikan [kredensial keamanan sementara](#) ke konsol AWS.

- API/CLI: AWS STS memverifikasi permintaan terhadap kebijakan kepercayaan peran untuk memastikan bahwa permintaan tersebut berasal dari entitas tepercaya (yaitu: akun Pengembangan). Setelah verifikasi, AWS STS mengembalikan [kredensial keamanan sementara](#) ke aplikasi.
5. Kredensial sementara mengizinkan akses ke sumber daya AWS:
- Konsol AWS: Konsol AWS menggunakan kredensial sementara atas nama pengguna untuk semua tindakan konsol berikutnya, dalam hal ini, untuk membaca dan menulis ke bucket `productionapp`. 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 bucket `productionapp`. 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.

Informasi selengkapnya

Untuk informasi lebih lanjut, lihat hal berikut:

- [Tutorial IAM: Delegasikan akses AWS ke akun yang menggunakan peran IAM](#)

Menyediakan akses untuk nonAWS beban kerja

[Peran IAM](#) adalah objek dalam AWS Identity and Access Management (IAM) yang ditugaskan [izin](#). Saat Anda [menggambil peran tersebut](#) menggunakan identitas IAM atau identitas dari luar AWS, kredensial keamanan sementara untuk sesi peran Anda akan diberikan. Anda mungkin memiliki beban kerja yang berjalan di pusat data atau infrastruktur lain di luar AWS yang perlu mengakses AWS sumber daya Anda. Alih-alih membuat, mendistribusikan, dan mengelola kunci akses jangka panjang, Anda dapat menggunakan AWS Identity and Access Management Roles Anywhere (IAM Roles Anywhere) untuk mengotentikasi nonAWS beban kerja Anda. IAM Roles Anywhere menggunakan sertifikat X.509 dari otoritas sertifikat (CA) Anda untuk mengautentikasi identitas dan menyediakan akses Layanan AWS dengan aman dengan kredensial sementara yang disediakan oleh peran IAM.

Untuk menggunakan IAM Roles Anywhere, Anda menyiapkan CA menggunakan [AWS Private Certificate Authority](#) atau menggunakan CA dari infrastruktur PKI Anda sendiri. Setelah menyiapkan CA, Anda membuat objek di IAM Roles Anywhere disebut trust anchor untuk membangun

kepercayaan antara IAM Roles Anywhere dan CA Anda untuk otentikasi. Anda kemudian dapat mengonfigurasi peran IAM yang ada, atau membuat peran baru yang mempercayai layanan IAM Roles Anywhere. Ketika nonAWS beban kerja Anda mengautentikasi dengan IAM Roles Anywhere menggunakan trust anchor, mereka bisa mendapatkan kredensi sementara untuk peran IAM Anda untuk mengakses AWS sumber daya Anda.

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 Peran IAM.

Menyediakan Akun AWS akses ke pihak ketiga

Ketika pihak ketiga memerlukan akses ke sumber daya AWS organisasi, Anda dapat menggunakan peran untuk mendelegasikan akses ke mereka. Misalnya, pihak ketiga mungkin menyediakan layanan untuk mengelola sumber daya AWS Anda. Dengan peran IAM, Anda dapat memberi pihak ketiga tersebut akses ke AWS sumber daya Anda tanpa berbagi kredensi AWS keamanan. Sebaliknya, pihak ketiga dapat mengakses AWS sumber daya Anda dengan mengasumsikan peran yang Anda buat di sumber daya yang 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 menetapkan Akun AWS ID mereka sebagai penanggung jawab ketika Anda menetapkan kebijakan kepercayaan untuk peran tersebut.
- ID eksternal untuk dikaitkan secara unik dengan peran tersebut. ID eksternal dapat berupa pengidentifikasi yang hanya Anda dan pihak ketiga ketahui. 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. Untuk informasi lebih lanjut tentang ID eksternal, lihat [Cara menggunakan ID eksternal saat memberikan akses ke sumber daya AWS Anda kepada pihak ketiga](#).
- Izin yang diperlukan pihak ketiga untuk bekerja dengan sumber daya AWS 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

Ketika Anda memberikan akses kepada pihak ketiga ke sumber daya AWS, mereka dapat mengakses semua sumber daya yang Anda tentukan dalam kebijakan. Penggunaan sumber daya Anda oleh mereka dibebankan kepada Anda. Pastikan bahwa Anda membatasi penggunaan sumber daya Anda dengan tepat.

Cara menggunakan ID eksternal saat memberikan akses ke sumber daya AWS Anda kepada pihak ketiga

Terkadang, Anda perlu memberi pihak ketiga akses ke sumber daya AWS (mendelegasikan akses). Satu aspek penting dari skenario ini adalah ID Eksternal, informasi opsional yang dapat Anda gunakan dalam kebijakan kepercayaan peran IAM untuk menentukan siapa yang dapat mengasumsikan peran tersebut.

 Important

AWS tidak menganggap ID eksternal sebagai rahasia. Setelah membuat rahasia seperti pasangan access key atau kata sandi di AWS, Anda tidak dapat melihatnya lagi. ID eksternal untuk peran dapat dilihat oleh siapa pun yang memiliki izin untuk melihat peran tersebut.

Dalam lingkungan multipenyewa di mana Anda mendukung beberapa pelanggan dengan AWS akun 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 menyediakan ID eksternal saat Anda mengambil peran, gunakan AWS CLI atau API AWS untuk mengambil peran tersebut. Untuk informasi lebih lanjut, lihat operasi API STS [AssumeRole](#), atau operasi CLI STS [assume-role](#).

Misalnya, katakanlah bahwa Anda memutuskan untuk merekrut perusahaan pihak ketiga bernama Example Corp untuk memantau Akun AWS dan membantu mengoptimalkan biaya. Untuk melacak

belanja harian Anda, Example Corp perlu mengakses sumber daya AWS Anda. Example Corp juga memantau banyak akun AWS lain untuk nasabah lain.

Jangan memberi Example Corp akses ke pengguna IAM dan kredensi jangka panjangnya pada AWS akun Anda. Sebaliknya, gunakan peran IAM dan kredensial keamanan sementara. Peran IAM memberikan mekanisme untuk memperbolehkan pihak ketiga mengakses AWS sumber daya Anda tanpa perlu berbagi kredensi jangka panjang (seperti kunci akses pengguna IAM).


Anda dapat menggunakan peran IAM untuk membangun hubungan terpercaya antara akun Akun AWS Example Corp. Setelah hubungan ini terjalin, anggota akun Example Corp dapat memanggil AWS Security Token Service [AssumeRole](#) API untuk mendapatkan kredensial keamanan sementara. Anggota Example Corp kemudian dapat menggunakan kredensial tersebut untuk mengakses sumber daya AWS di akun Anda.

 Note

Untuk informasi lebih lanjut tentang AssumeRole dan operasi AWS API lainnya yang dapat Anda panggil untuk mendapatkan kredensial keamanan sementara, lihat [Meminta kredensial keamanan sementara](#).

Berikut rincian skenario ini secara lebih detail:

1. Anda merekrut Example Corp, sehingga mereka membuat pengenalan pelanggan unik untuk Anda. Mereka memberi Anda. unik pelanggan 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, selama itu 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'. Example Corp harus memberikan ExternalId nilai kepada setiap pelanggan. Yang krusial adalah itu harus dihasilkan oleh Example Corp dan bukan pelanggan mereka untuk memastikan setiap ID eksternal.

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 contoh 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 sumber daya AWS Anda, seseorang dari perusahaan memanggil API AWS `sts:AssumeRole`. Panggilan mencakup ARN peran yang akan diasumsikan dan `ExternalId` parameter yang sesuai dengan ID pelanggan mereka.

Jika permintaan tersebut berasal dari seseorang yang menggunakan Example Corp Akun AWS, dan jika ARN peran dan ID eksternal sudah benar, permintaan tersebut berhasil. Kemudian, itu menyediakan kredensial keamanan sementara yang dapat digunakan oleh Example Corp untuk mengakses sumber daya AWS yang diizinkan oleh 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.

Mengapa menggunakan ID eksternal?

Dalam istilah abstrak, ID eksternal memperbolehkan pengguna yang mengasumsikan 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 mencegah [Masalah confused deputy](#).

Kapan saya harus menggunakan ID eksternal?

Gunakan ID eksternal dalam situasi-situasi berikut:

- Anda adalah Akun AWS pemilik dan Anda telah mengonfigurasi suatu peran untuk pihak ketiga yang mengakses orang lain Akun AWS selain 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`.

Menyediakan akses ke layanan AWS

Banyak layanan AWS yang mengharuskan Anda untuk menggunakan peran guna mengendalikan hal yang dapat diakses layanan tersebut. Peran yang diasumsikan layanan untuk melakukan tindakan atas nama Anda disebut [peran layanan](#). Ketika peran melayani tujuan khusus untuk layanan, peran dapat dikategorikan sebagai [peran layanan untuk instans EC2](#), atau [peran yang berkaitan dengan 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 guna mendelegasikan akses ke layanan yang ditawarkan oleh AWS, lihat [Membuat peran untuk mendelegasikan izin ke layanan AWS](#).

Masalah confused deputy

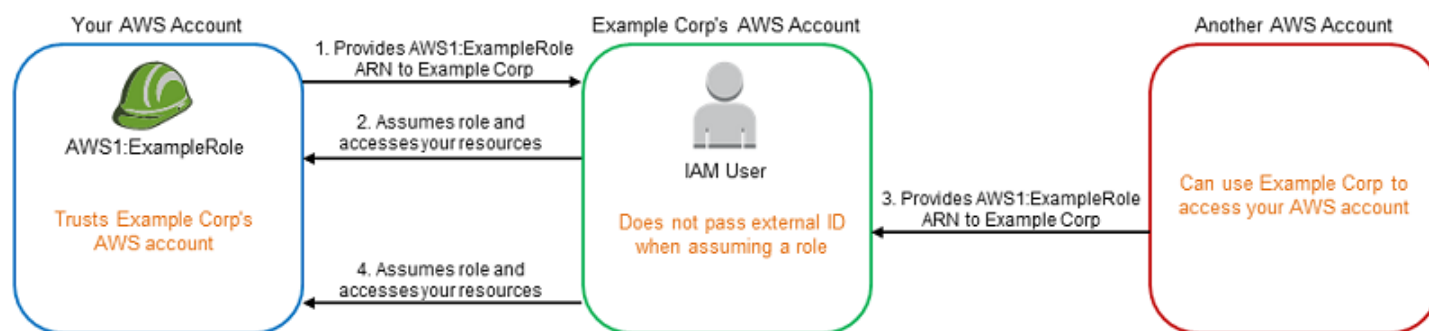
Masalah deputy yang bingung adalah masalah keamanan di mana entitas yang tidak memiliki izin untuk melakukan tindakan dapat memaksa entitas yang lebih istimewa untuk melakukan tindakan. Untuk mencegah hal ini, AWS menyediakan alat yang membantu Anda melindungi akun Anda jika Anda memberi pihak ketiga (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, 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. 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, informasi 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 deputy yang membingungkan.

Pada tahun AWS, peniruan lintas layanan dapat mengakibatkan masalah wakil yang membingungkan. Peniruan identitas lintas layanan dapat terjadi ketika satu layanan (layanan pemanggil) memanggil layanan lain (layanan yang dipanggil). Layanan pemanggil dapat dimanipulasi menggunakan izinnya untuk bertindak pada sumber daya pelanggan lain dengan cara yang seharusnya tidak dilakukannya kecuali bila memiliki izin untuk mengakses.

Pencegahan Deputy Bingung Lintas Akun

Diagram berikut menggambarkan masalah wakil kebingungan lintas akun.



Skenario ini mengasumsikan hal berikut:

- AWS1 adalah milikmu Akun AWS.
- AWS1: 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 `ExampleRole: keAWSExample Corp`.
2. Contoh Corp menggunakan peran ARN itu untuk mendapatkan kredensial 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 `ExampleRole: untukAWSExample Corp` untuk digunakan. Agaknya pelanggan lain belajar atau menebak AWS1: ExampleRole, yang bukan rahasia.
4. Ketika pelanggan lain meminta Example Corp untuk mengakses AWS sumber daya di (apa yang diklaimnya) akunnya, Example Corp menggunakan AWS1: 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

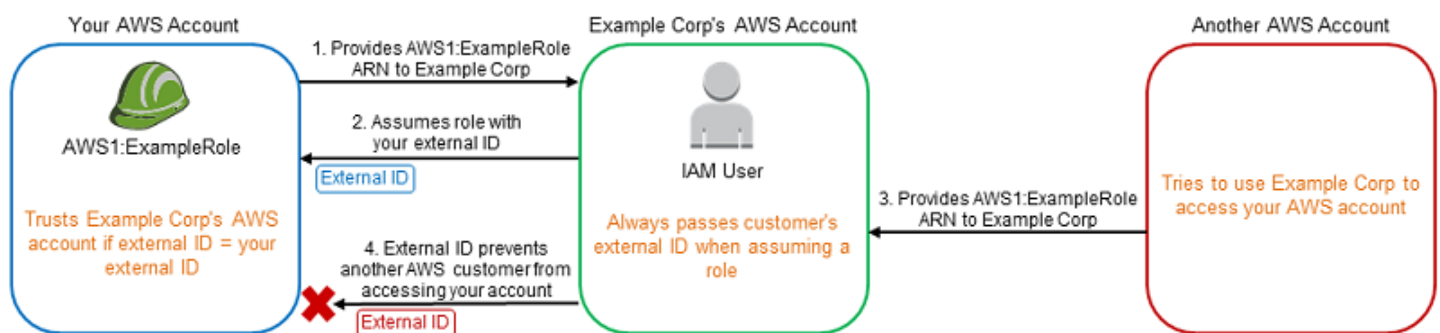
di antara pelanggan Example Corp, mereka adalah nilai yang masuk akal untuk digunakan bagi ID eksternal.

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:

```
{
  "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 IAM 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 `ExampleRole: keAWSExample Corp`.
2. Saat Example Corp menggunakan ARN peran itu untuk mengambil AWS peran `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 kredensi 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 `ExampleRole: untukAWSExample Corp` untuk digunakan.
4. Tapi kali ini, ketika Example Corp mencoba untuk mengambil peran `AWS1: 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 `AWS1: 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.

Pencegahan confused deputy lintas layanan

Sebaiknya gunakan kunci konteks kondisi

[aws:SourceArn](#), [aws:SourceAccount](#), [aws:SourceOrgID](#), atau [aws:SourceOrgPaths](#) global dalam kebijakan berbasis sumber daya untuk membatasi izin yang dimiliki layanan ke sumber daya tertentu. Gunakan [aws:SourceArn](#) untuk mengaitkan hanya satu sumber daya dengan akses lintas layanan. Gunakan [aws:SourceAccount](#) untuk membiarkan sumber daya apa pun di akun itu dikaitkan dengan penggunaan lintas layanan. Gunakan [aws:SourceOrgID](#) untuk memungkinkan sumber daya apa pun dari akun apa pun dalam suatu organisasi dikaitkan dengan penggunaan lintas layanan. Gunakan [aws:SourceOrgPaths](#) untuk mengaitkan sumber daya apa pun dari akun dalam AWS Organizations jalur dengan penggunaan lintas layanan. Untuk informasi selengkapnya tentang menggunakan dan memahami jalur, lihat [Memahami jalur AWS Organizations entitas](#).

Cara paling terperinci untuk melindungi dari masalah wakil yang membingungkan adalah dengan menggunakan kunci konteks kondisi [aws:SourceArn](#) global dengan ARN penuh sumber daya dalam kebijakan berbasis sumber daya Anda. Jika Anda tidak mengetahui ARN lengkap sumber daya atau jika Anda menentukan beberapa sumber daya, gunakan kunci konteks kondisi

`aws:SourceArn` global dengan wildcard (*) untuk bagian ARN yang tidak diketahui. Misalnya, `arn:aws:service:*:123456789012:*`.

Jika `aws:SourceArn` nilainya tidak berisi ID akun, seperti ARN bucket Amazon S3, Anda harus menggunakan keduanya `aws:SourceAccount` dan `aws:SourceArn` untuk membatasi izin.

Untuk melindungi dari masalah wakil yang membingungkan dalam skala besar, gunakan `aws:SourceOrgID` atau kunci konteks kondisi `aws:SourceOrgPaths` global dengan ID organisasi atau jalur organisasi sumber daya dalam kebijakan berbasis sumber daya Anda. Kebijakan yang menyertakan `aws:SourceOrgID` atau `aws:SourceOrgPaths` kunci akan secara otomatis menyertakan akun yang benar dan Anda tidak perlu memperbarui kebijakan secara manual saat menambahkan, menghapus, atau memindahkan akun di organisasi Anda.

Untuk [kebijakan kepercayaan non-service-linked](#) peran, setiap layanan dalam kebijakan kepercayaan telah melakukan `iam:PassRole` tindakan untuk memverifikasi bahwa peran tersebut berada di akun yang sama dengan layanan panggilan. Akibatnya, menggunakan `aws:SourceAccount`, `aws:SourceOrgID`, atau `aws:SourceOrgPaths` dengan kebijakan kepercayaan tersebut tidak diperlukan. Menggunakan `aws:SourceArn` dalam kebijakan kepercayaan memungkinkan Anda menentukan sumber daya peran yang dapat diasumsikan atas nama, seperti ARN fungsi Lambda. Beberapa kebijakan Layanan AWS penggunaan `aws:SourceAccount` dan `aws:SourceArn` kepercayaan untuk peran yang baru dibuat, tetapi menggunakan kunci tidak diperlukan untuk peran yang ada di akun Anda.

Note

Layanan AWS yang terintegrasi dengan AWS Key Management Service menggunakan hibah kunci KMS tidak mendukung kunci `aws:SourceArn`, `aws:SourceAccount`, `aws:SourceOrgID`, atau `aws:SourceOrgPaths` kondisi. Penggunaan kunci kondisi ini dalam kebijakan kunci KMS akan menghasilkan perilaku yang tidak terduga jika kunci juga digunakan oleh Layanan AWS melalui hibah kunci KMS.

Pencegahan deputy kebingungan lintas layanan untuk AWS Security Token Service

Banyak AWS layanan mengharuskan Anda menggunakan peran untuk memungkinkan layanan mengakses sumber daya layanan lain atas nama Anda. Peran yang diasumsikan layanan untuk melakukan tindakan atas nama Anda disebut [peran layanan](#). Peran memerlukan dua kebijakan: kebijakan kepercayaan peran yang menentukan prinsipal yang diizinkan untuk mengambil peran

dan kebijakan izin yang menentukan apa yang dapat dilakukan dengan peran tersebut. Kebijakan kepercayaan peran adalah satu-satunya jenis kebijakan berbasis sumber daya di IAM. Lainnya Layanan AWS memiliki kebijakan berbasis sumber daya, seperti kebijakan bucket Amazon S3.

Ketika layanan mengambil peran atas nama Anda, kepala layanan harus diizinkan untuk melakukan `sts:AssumeRole` tindakan dalam kebijakan kepercayaan peran. Saat layanan memanggil `sts:AssumeRole`, AWS STS mengembalikan sekumpulan kredensial keamanan sementara yang digunakan prinsipal layanan untuk mengakses sumber daya yang diizinkan oleh kebijakan izin peran. Saat layanan mengambil peran di akun Anda, Anda dapat menyertakan kunci konteks kondisi `aws:SourceArn`, `aws:SourceAccount`, `aws:SourceOrgID`, atau `aws:SourceOrgPaths` global dalam kebijakan kepercayaan peran Anda untuk membatasi akses ke peran hanya untuk permintaan yang dihasilkan oleh sumber daya yang diharapkan.

Misalnya, di AWS Systems Manager Incident Manager, Anda harus memilih peran untuk mengizinkan Manajer Insiden menjalankan dokumen otomatisasi Systems Manager atas nama Anda. Dokumen otomatisasi dapat mencakup rencana respons otomatis untuk insiden yang diprakarsai oleh CloudWatch alarm atau peristiwa. EventBridge Dalam contoh kebijakan kepercayaan peran berikut, Anda dapat menggunakan kunci `aws:SourceArn` kondisi untuk membatasi akses ke peran layanan berdasarkan ARN catatan insiden. Hanya catatan insiden yang dibuat dari sumber daya rencana respons `myresponseplan` yang dapat menggunakan peran ini.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {
      "Service": "ssm-incidents.amazonaws.com"
    },
    "Action": "sts:AssumeRole",
    "Condition": {
      "ArnLike": {
        "aws:SourceArn": "arn:aws:ssm-incidents:*:111122223333:incident-
record/myresponseplan/*"
      }
    }
  }
}
```

Note

Tidak semua integrasi layanan dengan AWS STS dukungan `aws:SourceArn`, `aws:SourceAccount`, `aws:SourceOrgID`, atau kunci `aws:SourceOrgPaths` kondisi. Penggunaan kunci ini dalam kebijakan kepercayaan IAM dengan integrasi yang tidak didukung dapat mengakibatkan perilaku yang tidak terduga.

Menyediakan 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 AWS kredensial 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, kami sarankan 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 \(bukan peran\)](#).

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 kredensi 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 SDK [AWS Seluler untuk iOS dan SDK Seluler untuk Android dan AWS Fire OS untuk membuat identitas unik bagi pengguna dan](#) mengautentikasi mereka untuk akses aman ke sumber daya Anda. AWS 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 [Menggunakan 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 behind-the-scenes pekerjaan 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 penggunaan federasi OIDC menggunakan salah satu layanan ini, lihat [Federasi OIDC](#)

Pengguna federasi dengan SAML 2.0

Jika organisasi Anda sudah menggunakan paket perangkat lunak penyedia identitas yang mendukung SAMP 2.0 (Security Assertion Markup Language 2.0), Anda dapat membuat kepercayaan antara organisasi Anda sebagai penyedia identitas (iDP) dan sebagai penyedia layanan. AWS Anda kemudian dapat menggunakan SAMP untuk menyediakan pengguna Anda dengan federated single-sign on (SSO) ke AWS Management Console 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

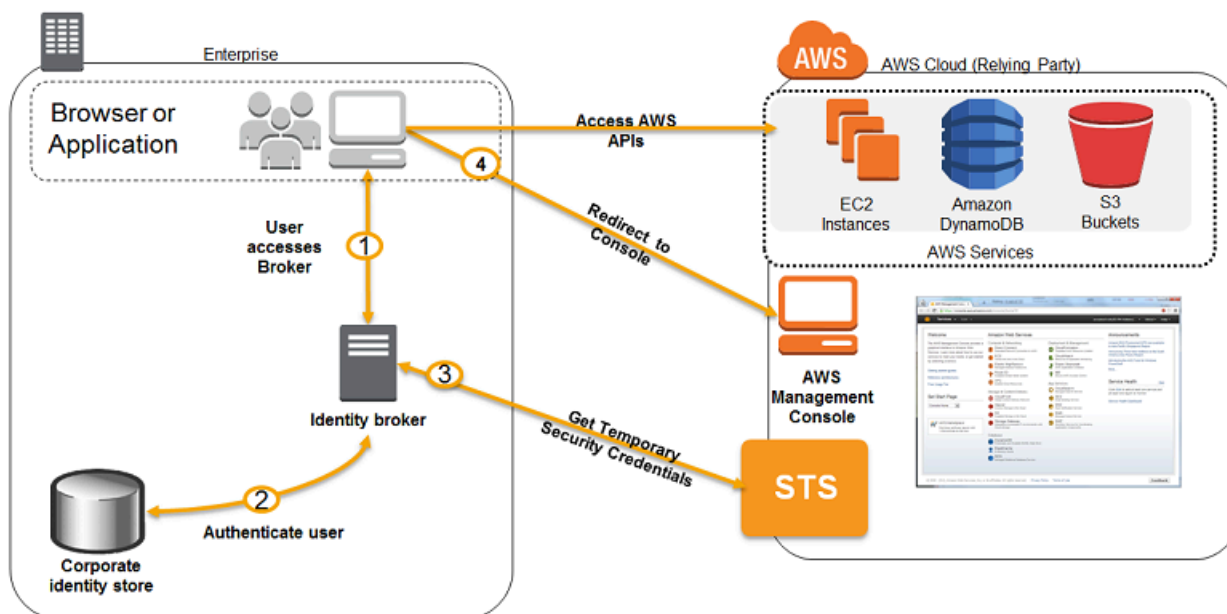
Jika penyimpanan identitas tidak kompatibel dengan SAML 2.0, maka Anda dapat membuat aplikasi broker identitas khusus untuk melakukan fungsi yang serupa. Aplikasi broker mengotentikasi pengguna, meminta kredensi sementara untuk pengguna dari AWS, dan kemudian menyediakannya kepada pengguna untuk mengakses sumber daya. AWS

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 kustom. 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

husus), 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 [Mengontrol izin untuk kredensial keamanan sementara](#).) Panggilan mengembalikan kredensial 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 Konsol AWS Manajemen (yang disebut sebagai sistem masuk tunggal).

Untuk informasi tentang membuat kredensial keamanan sementara, lihat [Meminta kredensial keamanan sementara](#). Untuk informasi selengkapnya tentang pengguna federasi yang mendapatkan akses ke AWS Management Console, lihat [Mengaktifkan pengguna gabungan SAML 2.0 untuk mengakses AWS Management Console](#).

Menggunakan peran terkait layanan

Peran terkait layanan adalah jenis IAM role unik yang terhubung langsung ke sebuah layanan AWS. Peran terkait layanan ditentukan sebelumnya oleh layanan dan mencakup semua izin yang diperlukan layanan untuk memanggil layanan AWS lainnya 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. Terlepas dari metodenya, peran terkait layanan menyederhanakan proses penyiapan 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 [IAM role](#) yang diambil oleh layanan untuk melakukan tindakan atas nama Anda. Administrator IAM dapat membuat, memodifikasi, dan menghapus peran layanan dari dalam IAM. Untuk informasi lebih lanjut, lihat [Membuat peran untuk mendelegasikan izin keLayanan AWS](#) di Panduan Pengguna IAM. Peran terkait layanan adalah jenis peran layanan yang ditautkan keLayanan AWS. Layanan dapat menggunakan peran untuk melakukan tindakan atas nama Anda. Peran terkait layanan muncul diAkun 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 mereka. Ini melindungi sumber daya Anda karena Anda tidak dapat secara tidak sengaja menghapus izin untuk mengakses sumber daya.

Tip

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 Pilih Ya dengan tautan untuk melihat dokumentasi peran terkait layanan untuk layanan tersebut.

Izin peran terkait layanan

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 prinsipal layanan, karena peka huruf besar kecil dan formatnya dapat beragam di berbagai layanan AWS. Untuk melihat prinsipal layanan untuk suatu layanan, lihat dokumentasi peran yang terkait dengan layanan.

Untuk memungkinkan entitas IAM membuat peran terkait layanan tertentu

Tambahkan kebijakan berikut ke entitas IAM yang perlu membuat peran tertaut-layanan.

```
{
  "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 memungkinkan entitas IAM meneruskan peran yang ada ke layanan

Beberapa layanan AWS memungkinkan Anda memberikan peran yang sudah ada ke layanan tersebut, 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 [Memberi izin kepada pengguna untuk meneruskan peran ke layanan AWS](#).

```

{
  "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 itu dapat menggunakan izinnya sendiri untuk memanggil yang lain AWS layanan. 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 instans Amazon RDS DB, [peran terkait layanan untuk RDS](#) secara otomatis dibuat jika salah satu belum ada. Peran terkait layanan ini memungkinkan RDS memanggil Amazon EC2, Amazon SNS, Amazon CloudWatch Log, dan Amazon Kinesis 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, Amazon CloudWatch Log log, dan sumber daya Amazon Kinesis dengan memanggil RDS, karena RDS akan menggunakan peran terkait layanannya untuk mengakses sumber daya tersebut.

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 akun AWS saya](#).

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 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 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

Peran terkait layanan diperhitungkan terhadap Anda [Peran IAM dalam sebuah Akun AWS](#) limit, 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 tertaut layanan (konsol)

1. Masuk ke AWS Management Console 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 AWS Layanan tipe peran.
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 serangkaian 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 samping 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 [Menandai sumber daya IAM](#).
9. Tinjau peran dan kemudian 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 (API AWS)

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 API AWS untuk membuat peran terkait layanan menggunakan kebijakan kepercayaan dan kebijakan inline yang diperlukan layanan untuk mengambil peran tersebut.

Untuk membuat peran terkait layanan (AWSAPI)

Gunakan panggilan API [CreateServiceLinkedRole](#). Dalam permintaan, sebutkan nama layanan dari `SERVICE_NAME_URL .amazonaws .com`.

Misalnya, untuk membuat peran terkait layanan Lex Bots, gunakan `lex .amazonaws .com`.

Mengedit peran yang ditautkan ke 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 tertaut layanan (konsol)

1. Di panel navigasi konsol IAM pilih Peran.
2. Pilih nama peran yang akan dimodifikasi.
3. Di sisi kanan jauh dari Deskripsi peran, pilih Sunting.
4. Masukkan deskripsi baru di kotak, dan pilih Simpan.

Mengedit deskripsi peran tertaut layanan (AWS CLI)

Anda dapat menggunakan perintah IAM dari AWS CLI untuk mengedit deskripsi peran tertaut 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 API AWS 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:

AWSAPI: [GetRole](#)

2. Untuk memperbarui deskripsi peran, hubungi operasi berikut, dan sebutkan nama (dan deskripsi opsional) peran tersebut:

AWSAPI: [UpdateRole](#)

Menghapus peran yang ditautkan ke layanan

Metode yang Anda gunakan untuk membuat peran terkait layanan tergantung pada layanannya. 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.

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 penghapusan peran tertaut-layanan, pilih tautan Ya untuk melihat dokumentasi peran tertaut-layanan untuk layanan itu.

Jika layanan tidak mendukung penghapusan peran, Anda dapat menghapus peran terkait layanan dari konsol IAM, API, atau AWS CLI. Jika Anda tidak perlu lagi menggunakan fitur atau layanan yang memerlukan peran terkait layanan, kami menyarankan Anda menghapus peran tersebut. Dengan begitu, Anda tidak memiliki entitas yang tidak digunakan yang tidak dipantau atau dipelihara secara aktif. Namun, Anda harus membersihkan peran terkait layanan sebelum dapat menghapusnya.

Membersihkan peran tertaut layanan

Sebelum Anda dapat menggunakan IAM untuk menghapus peran terkait layanan, Anda harus mengonfirmasi terlebih dahulu bahwa peran tersebut tidak memiliki sesi aktif dan menghapus sumber daya yang digunakan oleh peran tersebut.

Untuk memastikan peran tertaut layanan memiliki sesi aktif di konsol IAM

1. Masuk ke AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Di panel navigasi konsol IAM, pilih Peran. Kemudian pilih nama (bukan kotak centang) dari peran yang ditautkan ke layanan.
3. Di halaman Ringkasan untuk peran yang dipilih, pilih tab Access Advisor.
4. Di tab Penasihat Akses, Tinjau aktivitas terbaru untuk peran terkait layanan.

Note

Jika Anda tidak yakin apakah layanan menggunakan peran tertaut-layanan, Anda dapat mencoba untuk menghapus peran tersebut. Jika layanan menggunakan peran tersebut, maka penghapusan gagal dan Anda dapat melihat wilayah tempat peran tersebut digunakan. Jika peran tersebut sedang digunakan, Anda harus menunggu hingga sesi

ini berakhir sebelum dapat menghapus peran tersebut. Anda tidak dapat mencabut sesi untuk peran terkait layanan.

Untuk menghapus sumber daya yang digunakan oleh peran terkait layanan

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 penghapusan peran tertaut-layanan, pilih tautan Ya untuk melihat dokumentasi peran tertaut-layanan untuk layanan itu. Lihat dokumentasi layanan tersebut untuk mempelajari cara menghapus sumber daya yang digunakan oleh peran tertaut-layanan Anda.

Menghapus peran yang ditautkan ke layanan (konsol)

Anda dapat menggunakan konsol IAM untuk menghapus peran terkait layanan.

Untuk menghapus peran terkait layanan (konsol)

1. Masuk ke AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Di panel navigasi konsol IAM, pilih Peran. Lalu pilih kotak centang di sebelah nama peran yang ingin Anda hapus, bukan nama atau baris itu sendiri.
3. Untuk Tindakan peran di bagian atas halaman, pilih Delete (Hapus).
4. Dalam kotak dialog konfirmasi, tinjau informasi yang terakhir diakses, yang menunjukkan kapan terakhir kali setiap peran yang dipilih mengakses layanan AWS. Ini membantu Anda mengonfirmasi apakah peran tersebut saat ini aktif. Jika Anda ingin melanjutkan, pilih Ya, Hapus untuk mengirimkan peran terkait layanan untuk penghapusan.
5. Perhatikan pemberitahuan konsol IAM untuk memantau kemajuan penghapusan peran terkait layanan. Karena penghapusan peran terkait layanan IAM bersifat asinkron, setelah Anda mengirimkan peran tersebut untuk dihapus, tugas penghapusan dapat berhasil atau gagal.
 - Jika tugas berhasil, maka peran tersebut dihapus dari daftar dan pemberitahuan keberhasilan muncul di bagian atas halaman.
 - Jika tugas tersebut gagal, Anda dapat memilih Lihat rincian atau Lihat Sumber Daya dari pemberitahuan untuk mempelajari alasan penghapusan gagal. Jika penghapusan gagal karena peran tersebut sedang menggunakan sumber daya layanan, maka pemberitahuan

mencakup daftar sumber daya, jika layanan tersebut mengembalikan informasi tersebut. Anda kemudian dapat [membersihkan sumber daya](#) dan mengirimkan penghapusan lagi.

Note

Anda mungkin harus mengulangi proses ini beberapa kali, tergantung pada informasi yang dikembalikan oleh layanan. Misalnya, peran terkait layanan Anda mungkin menggunakan enam sumber daya dan layanan Anda mungkin akan mengembalikan informasi mengenai lima di antaranya. Jika Anda membersihkan lima sumber daya dan mengirim peran untuk penghapusan lagi, penghapusan tersebut gagal dan layanan melaporkan sumber daya yang tersisa. Layanan mungkin mengembalikan semua sumber daya, beberapa di antaranya, atau tidak melaporkan sumber daya apa pun.

- Jika tugas gagal dan pemberitahuan tidak mencakup daftar sumber daya, maka layanan mungkin tidak mengembalikan informasi tersebut. Untuk mempelajari cara membersihkan sumber daya untuk layanan tersebut, lihat [AWS layanan yang bekerja dengan IAM](#). Cari layanan Anda dalam tabel, dan pilih tautan Ya untuk melihat dokumentasi peran terkait layanan untuk layanan tersebut

Menghapus peran tertaut layanan (AWS CLI)

Anda dapat menggunakan perintah IAM dari AWS CLI untuk menghapus peran tertaut layanan.

Untuk menghapus peran tertaut layanan (AWS CLI)

1. Jika Anda mengetahui nama peran terkait layanan yang ingin Anda hapus, masukkan perintah berikut untuk mencantumkan peran di akun Anda:

```
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. Karena peran terkait layanan tidak dapat dihapus jika sedang digunakan atau memiliki sumber daya terkait, Anda harus mengirimkan permintaan penghapusan. Permintaan tersebut dapat ditolak jika syarat ini tidak terpenuhi. Anda harus menangkap `deletion-task-id` dari tanggapan untuk memeriksa status tugas penghapusan. Ketik perintah berikut untuk mengirimkan permintaan penghapusan peran yang terhubung dengan layanan:

```
aws iam delete-service-linked-role --role-name role-name
```

3. Masukkan perintah berikut untuk memeriksa status tugas penghapusan:

```
aws iam get-service-linked-role-deletion-status --deletion-task-id deletion-task-id
```

Kemungkinan status tugas penghapusan adalah NOT_STARTED, IN_PROGRESS, SUCCEEDED, atau FAILED. Jika penghapusan gagal, panggilan akan mengembalikan alasan kegagalan panggilan sehingga Anda dapat memecahkan masalah. Jika penghapusan gagal karena peran tersebut sedang menggunakan sumber daya layanan, maka pemberitahuan mencakup daftar sumber daya, jika layanan tersebut mengembalikan informasi tersebut. Anda kemudian dapat [membersihkan sumber daya](#) dan mengirimkan penghapusan lagi.

Note

Anda mungkin harus mengulangi proses ini beberapa kali, tergantung pada informasi yang dikembalikan oleh layanan. Misalnya, peran terkait layanan Anda mungkin menggunakan enam sumber daya dan layanan Anda mungkin akan mengembalikan informasi mengenai lima di antaranya. Jika Anda membersihkan lima sumber daya dan mengirim peran untuk penghapusan lagi, penghapusan tersebut gagal dan layanan melaporkan sumber daya yang tersisa. Layanan mungkin mengembalikan semua sumber daya, beberapa di antaranya, atau tidak melaporkan sumber daya apa pun. Untuk mempelajari cara membersihkan sumber daya untuk layanan yang tidak melaporkan sumber daya, lihat [AWS layanan yang bekerja dengan IAM](#). Cari layanan Anda dalam tabel, dan pilih tautan Ya untuk melihat dokumentasi peran terkait layanan untuk layanan tersebut

Menghapus peran terkait layanan (API AWS)

Anda dapat menggunakan API AWS untuk menghapus peran terkait layanan.

Untuk menghapus peran terkait layanan (API AWS)

1. Untuk mengirimkan permintaan penghapusan peran terkait layanan, hubungi [DeleteServiceLinkedRole](#). Pada permintaan itu, tentukan nama peran.

Karena peran tertaut layanan tidak dapat dihapus jika sedang digunakan atau memiliki sumber daya terkait, Anda harus mengirimkan permintaan penghapusan. Permintaan tersebut dapat ditolak jika syarat ini tidak terpenuhi. Anda harus menangkap `DeletionTaskId` dari tanggapan untuk memeriksa status tugas penghapusan.

2. Untuk memeriksa status penghapusan, panggil [GetServiceLinkedRoleDeletionStatus](#). Pada permintaan itu, tentukan `DeletionTaskId`.

Kemungkinan status tugas penghapusan adalah `NOT_STARTED`, `IN_PROGRESS`, `SUCCEEDED`, atau `FAILED`. Jika penghapusan gagal, panggilan akan mengembalikan alasan kegagalan panggilan sehingga Anda dapat memecahkan masalah. Jika penghapusan gagal karena peran tersebut sedang menggunakan sumber daya layanan, maka pemberitahuan mencakup daftar sumber daya, jika layanan tersebut mengembalikan informasi tersebut. Anda kemudian dapat [membersihkan sumber daya](#) dan mengirimkan penghapusan lagi.

Note

Anda mungkin harus mengulangi proses ini beberapa kali, tergantung pada informasi yang dikembalikan oleh layanan. Misalnya, peran terkait layanan Anda mungkin menggunakan enam sumber daya dan layanan Anda mungkin akan mengembalikan informasi mengenai lima di antaranya. Jika Anda membersihkan lima sumber daya dan mengirim peran untuk penghapusan lagi, penghapusan tersebut gagal dan layanan melaporkan sumber daya yang tersisa. Layanan mungkin mengembalikan semua sumber daya, beberapa di antaranya, atau tidak melaporkan sumber daya apa pun. Untuk mempelajari cara membersihkan sumber daya untuk layanan yang tidak melaporkan sumber daya, lihat [AWS layanan yang bekerja dengan IAM](#). Cari layanan Anda dalam tabel, dan pilih tautan Ya untuk melihat dokumentasi peran terkait layanan untuk layanan tersebut

Membuat Peran IAM

Untuk membuat peran, Anda dapat menggunakan [AWS Management Console](#), [Alat untuk Windows PowerShell](#), atau [IAM API](#). [AWS CLI](#)

Jika Anda menggunakan [AWS Management Console](#), wizard akan memandu Anda melalui langkah-langkah untuk membuat peran. Wizard memiliki langkah yang sedikit berbeda tergantung pada

apakah Anda membuat peran untuk AWS layanan layanan, untuk layanan Akun AWS, untuk layanan, atau untuk pengguna federasi.

Topik

- [Membuat peran untuk mendelegasikan izin kepada pengguna IAM](#)
- [Membuat peran untuk mendelegasikan izin ke layanan AWS](#)
- [Membuat peran untuk Penyedia Identitas pihak ketiga \(federasi\)](#)
- [Membuat peran menggunakan kebijakan kepercayaan khusus \(konsol\)](#)
- [Contoh kebijakan untuk mendelegasikan akses](#)

Membuat peran untuk mendelegasikan izin kepada pengguna IAM

Anda dapat menggunakan peran IAM untuk mendelegasikan akses ke sumber daya Anda AWS. 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 tepercaya 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 memberikan kredensial keamanan sementara yang memungkinkan akses ke sumber daya AWS 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 [Cara menggunakan ID eksternal saat memberikan akses ke sumber daya AWS Anda kepada 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 [Membuat peran untuk mendelegasikan izin ke layanan AWS](#).

Membuat peran IAM (konsol)

Anda dapat menggunakan AWS Management Console 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, lihat [Contoh skenario menggunakan akun pengembangan dan produksi terpisah](#).

Untuk membuat peran (konsol)

1. Masuk ke AWS Management Console 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 AWS peran.
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 [Cara menggunakan ID eksternal saat memberikan akses ke sumber daya AWS Anda kepada 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 konsol AWS untuk beralih ke peran yang memiliki syarat `externalId` 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 AWS Management Console](#) 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 [Menggunakan otentikasi multi-faktor \(MFA\) di AWS](#)
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. Centang kotak di samping kebijakan izin yang Anda inginkan untuk dimiliki oleh siapa pun yang memegang 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 [Menandai sumber daya IAM](#).
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 tepercaya untuk beralih ke peran dalam konsol, atau mengasumsikan peran tersebut secara terprogram. Untuk informasi selengkapnya tentang langkah ini, lihat [Memberikan izin pengguna untuk berganti peran](#).

Membuat peran IAM () AWS CLI

Membuat peran dari AWS CLI mencakup beberapa langkah. Saat Anda menggunakan konsol untuk membuat peran, banyak langkah yang dilakukan untuk Anda, tetapi dengan AWS CLI 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 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 fitur lanjutan AWS.

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 Perintah AWS](#).

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 [Menggunakan otentikasi multi-faktor \(MFA\) di AWS](#).

```
{
  "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 prinsipal muncul di konsol karena AWS tidak 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.

```
{
  "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 [Memberikan izin pengguna untuk berganti peran](#).

Setelah Anda membuat peran dan memberinya izin untuk melaksanakan tugas AWS atau mengakses sumber daya AWS, setiap pengguna dalam akun 123456789012 dapat mengambil peran. Untuk informasi selengkapnya, lihat [Beralih ke peran IAM \(AWS CLI\)](#).

Membuat peran IAM (AWSAPI)

Membuat peran dari API AWS mencakup 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 (API AWS)

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 [Memberikan izin pengguna untuk berganti peran](#).

3. (Opsional) Tambahkan atribut khusus ke pengguna dengan melampirkan tag: [TagRole](#)

Untuk informasi selengkapnya, lihat [Mengelola tag pada pengguna 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 fitur lanjutan AWS.

Setelah Anda membuat peran dan memberinya izin untuk melaksanakan tugas AWS atau mengakses sumber daya AWS, Anda harus memberikan izin kepada setiap pengguna dalam akun agar mereka dapat mengambil peran. 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

Membuat peran untuk mendelegasikan izin ke layanan AWS

Banyak layanan AWS mengharuskan Anda menggunakan peran untuk memungkinkan layanan mengakses sumber daya dalam layanan lain atas nama Anda. Peran yang diasumsikan layanan untuk melakukan tindakan atas nama Anda disebut [peran layanan](#). Ketika peran memiliki tujuan khusus untuk suatu layanan, itu dikategorikan sebagai [peran layanan untuk instans EC2](#) (misalnya), atau [peran yang terkait dengan 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 menyetel `PassRole` 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 mengonfigurasi izin untuk memperbolehkan entitas IAM (pengguna atau peran) untuk membuat atau mengubah peran terkait layanan.

Note

ARN untuk peran terkait layanan mencakup prinsip layanan, yang ditunjukkan dalam kebijakan berikut sebagai *SERVICE-NAME*.amazonaws.com. Jangan mencoba menebak prinsipal layanan, karena peka huruf besar/kecil dan formatnya dapat bervariasi antar AWS layanan. Untuk melihat prinsipal layanan untuk suatu layanan, lihat dokumentasi peran yang terkait dengan layanan.

Untuk memungkinkan entitas IAM untuk 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.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iam:AttachRolePolicy",
        "iam:CreateRole",
        "iam:PutRolePolicy"
      ],
      "Resource": "arn:aws:iam::*:role/SERVICE-ROLE-NAME"
    }
  ]
}
```

Untuk memungkinkan entitas IAM membuat peran layanan apa pun

AWS merekomendasikan bahwa Anda hanya mengizinkan pengguna administratif untuk membuat peran layanan apa pun. Seseorang yang memiliki izin untuk membuat peran dan melampirkan kebijakan apa pun dapat meningkatkan izinnya sendiri. Sebagai gantinya, buat kebijakan yang memungkinkan mereka hanya membuat peran yang mereka butuhkan atau minta administrator membuat peran layanan atas nama mereka.

Untuk melampirkan kebijakan yang memungkinkan administrator mengakses seluruh kebijakan Anda Akun AWS, gunakan kebijakan [AdministratorAccess](#) AWS Sterkelola.

Untuk mengizinkan entitas IAM mengedit peran layanan

Tambahkan kebijakan berikut ke entitas IAM yang perlu mengubah peran layanan.

```
{
  "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 merekomendasikan bahwa 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 minta administrator menghapus peran layanan atas nama mereka.

Untuk melampirkan kebijakan yang memungkinkan administrator mengakses seluruh kebijakan Anda Akun AWS, gunakan kebijakan [AdministratorAccess](#) AWS terkelola.

Membuat peran untuk layanan AWS (konsol)


Anda dapat menggunakan AWS Management Console untuk membuat peran bagi 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.

Untuk membuat peran untuk Layanan AWS (konsol IAM)

1. Masuk ke AWS Management Console 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 jenis entitas Tepercaya, pilih Layanan AWS.

4. Untuk kasus Layanan atau 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 Izin, opsi bergantung pada kasus penggunaan yang Anda pilih:
 - Jika layanan menentukan izin untuk peran tersebut, Anda tidak dapat memilih kebijakan izin.
 - Pilih dari serangkaian kebijakan izin terbatas.
 - Pilih dari semua kebijakan izin.
 - Pilih kebijakan tanpa izin, buat kebijakan setelah peran dibuat, lalu lampirkan kebijakan ke peran.
7. (Opsional) Tetapkan [batas izin](#). Ini adalah fitur lanjutan yang tersedia untuk peran layanan, tetapi bukan peran tertaut layanan.
 - a. Buka bagian Setel 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 Peran, opsi bergantung 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.

 Important

Saat Anda memberi nama peran, perhatikan hal berikut:

- Nama peran harus unik di dalam diri AndaAkun AWS, dan tidak dapat dibuat unik berdasarkan kasus.

Misalnya, jangan membuat peran bernama keduanya **PRODRROLE** 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 tersebut 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 tersebut.
11. (Opsional) Untuk mengedit kasus penggunaan dan izin untuk peran, di 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 nilai kunci. Untuk informasi selengkapnya tentang penggunaan tanda di IAM, lihat [Menandai sumber daya IAM](#) di Panduan Pengguna IAM.
13. Tinjau peran lalu pilih Buat peran.

Membuat peran untuk layanan (AWS CLI)

Membuat peran dari AWS CLI mencakup beberapa langkah. Saat Anda menggunakan konsol untuk membuat peran, banyak langkah yang dilakukan untuk Anda, tetapi dengan AWS CLI 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 layanan AWS dari AWS CLI

1. [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
```

2. Lampirkan kebijakan izin terkelola ke peran: [aws attach-role-policy iam](#).

Misalnya, `attach-role-policy` perintah berikut melampirkan kebijakan AWS terkelola yang dinamai `ReadOnlyAccess` 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 sebaris untuk peran tersebut: `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: `aws iam put-role-permissions-boundary`](#)

Batas izin mengontrol izin maksimum yang dapat dimiliki sebuah peran. Batas izin adalah fitur lanjutan AWS.

Jika Anda akan menggunakan peran dengan Amazon EC2 atau AWS layanan lain yang menggunakan Amazon EC2, Anda harus menyimpan peran tersebut dalam profil instans. Profil instans adalah wadah untuk peran yang dapat dilampirkan ke instans Amazon EC2 saat diluncurkan. Profil instans hanya dapat berisi satu peran, dan batas tersebut tidak dapat ditingkatkan. Jika Anda membuat peran dengan menggunakan AWS Management Console, profil instans dibuat untuk Anda dengan nama sama seperti peran tersebut. Untuk informasi selengkapnya tentang profil instans, lihat [Menggunakan profil instans](#). Untuk informasi tentang cara meluncurkan instans EC2 dengan peran, lihat [Mengendalikan Akses ke Sumber Daya Amazon EC2](#) dalam Panduan Pengguna Amazon EC2 untuk Instans Linux.

Untuk membuat profil instans dan menyimpan peran di dalamnya (AWS CLI)

1. Buat profil instance: [aws iam create-instance-profile](#)
2. Tambahkan peran ke profil instance: [aws iam add-role-to-instance](#) -profile

Rangkaian perintah contoh AWS CLI 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 [Mengonfigurasi Antarmuka Baris Perintah AWS](#).

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.

```
{
  "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.

```
{
  "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 Mengonfigurasi Detail Instans jika Anda menggunakan konsol AWS. Jika Anda menggunakan `aws ec2 run-instances` perintah CLI, tentukan `--iam-instance-profile` parameter.

Membuat peran untuk layanan (API AWS)

Membuat peran dari API AWS mencakup 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 bagi layanan AWS (API AWS)

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 pengguna dengan melampirkan tag: [TagRole](#)

Untuk informasi selengkapnya, lihat [Mengelola tag pada pengguna 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 fitur lanjutan AWS.

Jika Anda akan menggunakan peran dengan Amazon EC2 atau AWS layanan lain yang menggunakan Amazon EC2, Anda harus menyimpan peran tersebut dalam profil instans. Profil contoh adalah wadah untuk peran. Profil instans hanya dapat berisi satu peran, dan batas tersebut tidak dapat ditingkatkan. Jika Anda membuat peran dengan di AWS Management Console, profil instans dibuat untuk Anda dengan nama sama seperti peran tersebut. Untuk informasi selengkapnya tentang profil instans, lihat [Menggunakan profil instans](#). Untuk informasi tentang cara meluncurkan instans EC2 dengan peran, lihat [Mengendalikan Akses ke Sumber Daya Amazon EC2](#) dalam Panduan Pengguna Amazon EC2 untuk Instans Linux.

Untuk membuat profil instans dan menyimpan peran di dalamnya (API AWS)

1. Buat profil instans: [CreateInstanceProfile](#)
2. Tambahkan peran ke profil instans: [AddRoleToInstanceProfile](#)

Membuat peran untuk Penyedia Identitas pihak ketiga (federasi)

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 dan federasi identitas](#).

Membuat peran untuk pengguna federasi (konsol)

Prosedur untuk membuat peran bagi pengguna federasi bergantung pada pilihan penyedia pihak ketiga Anda:

- Untuk OpenID Connect (OIDC), lihat [Buat peran untuk federasi OpenID Connect \(konsol\)](#)
- Untuk SAML 2.0, lihat [Buat peran untuk federasi SAMP 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 menciptakan 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 bagi federasi identitas (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.

```
{
```



```

"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.

```

{
  "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 `permpolicyforcognitofederation.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
\permpolicyforcognitofederation.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 menciptakan peran untuk OIDC](#).
- Untuk penyedia SAML, lihat [Prasyarat untuk membuat peran untuk SAML](#).

Untuk membuat peran untuk federasi identitas (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 pengguna dengan melampirkan tag: [TagRole](#)

Untuk informasi selengkapnya, lihat [Mengelola tag pada pengguna 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.

Buat peran untuk federasi OpenID Connect (konsol)

Anda dapat menggunakan penyedia identitas federasi OpenID Connect (OIDC) alih-alih membuat pengguna di penyedia identitas Anda. AWS Identity and Access Management 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, lihat [Penyedia dan federasi identitas](#).

Prasyarat untuk menciptakan peran untuk OIDC

Sebelum Anda dapat membuat peran untuk federasi OIDC, Anda harus terlebih dahulu menyelesaikan langkah-langkah prasyarat berikut.

Untuk mempersiapkan diri untuk menciptakan 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 dengan informasi penyedia. Saat melakukannya, penyedia memberi Anda aplikasi atau ID 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 [Buat penyedia identitas OpenID Connect \(OIDC\) di IAM](#).

Important

Jika Anda menggunakan IdP OIDC dari Google, Facebook, atau Amazon Cognito, jangan buat iDP IAM terpisah di file. AWS Management Console 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. Persiapkan kebijakan untuk peran yang akan diambil oleh pengguna yang diotentikasi IdP. 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 menyarankan Anda menggunakan [Amazon Cognito](#) untuk mengelola identitas. Dalam hal ini, gunakan kebijakan kepercayaan yang serupa dengan contoh ini.

```
{
  "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 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, itu *providerUrl* adalah URL. Contoh berikut termasuk metode untuk menentukan prinsipal 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 Nama Sumber Daya Amazon (ARN) dari IDP OIDC yang Anda buat, seperti contoh berikut: [Step 2](#)

```
"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. Pencocokan antara ID ini memastikan bahwa permintaan berasal dari aplikasi Anda.

Note

Peran IAM untuk kumpulan identitas Amazon Cognito mempercayai `cognito-identity.amazonaws.com` kepala layanan untuk mengambil peran tersebut. Peran jenis ini harus mengandung setidaknya satu kunci kondisi untuk membatasi kepala sekolah yang dapat mengambil peran tersebut.

Pertimbangan tambahan berlaku untuk kumpulan identitas Amazon Cognito yang mengambil peran IAM [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 bagi 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 autentikasi 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 spesifik untuk IDP. Peran untuk OIDC hanya dapat menentukan satu prinsipal. Oleh karena itu, jika aplikasi seluler memungkinkan pengguna untuk 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 with 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 with Amazon.

```
{
  "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.

```
{
  "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.

```

{
  "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.

```

{
  "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"}}
  ]}
}

```

Menciptakan peran untuk OIDC

Setelah Anda menyelesaikan prasyarat, Anda dapat membuat peran dalam IAM. Prosedur berikut menjelaskan cara membuat peran untuk federasi OIDC di AWS Management Console. Untuk membuat peran dari AWS CLI atau AWS API, lihat prosedur di [Membuat peran untuk Penyedia Identitas pihak ketiga \(federasi\)](#).

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 AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Pada panel navigasi, silakan pilih Peran lalu pilih Buat peran.
3. Pilih tipe peran OIDC.
4. Untuk penyedia Identity, pilih iDP untuk peran Anda:
 - Jika Anda ingin membuat peran untuk IDP web individual, pilih Login with 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 Cognito.

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 \(identitas gabungan\)](#) di Panduan Pengembang Amazon Cognito.

- Jika Anda ingin membuat peran untuk GitHub Tindakan, 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 AWS mengonfigurasi OIDC trust GitHub sebagai identitas federasi, lihat [GitHub Dokumen - Mengonfigurasi OpenID Connect di Amazon Web Services](#). Untuk informasi tentang praktik terbaik untuk membatasi akses untuk peran yang terkait dengan IDP IAM GitHub untuk, [Mengkonfigurasi peran untuk penyedia identitas GitHub OIDC](#) lihat di halaman ini.

5. Masukkan pengenal untuk aplikasi Anda. Label pengenal berubah berdasarkan penyedia yang Anda pilih:
 - Jika Anda ingin membuat peran untuk Login with 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 ID Kumpulan Identitas.
 - Jika Anda ingin membuat peran untuk GitHub Tindakan, masukkan detail berikut:
 - Untuk Audiens, pilih `sts.amazonaws.com`.
 - Untuk GitHub organisasi, masukkan nama GitHub organisasi Anda. 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, itu default ke wildcard (). *
 - (Opsional) Untuk GitHub cabang, masukkan nama GitHub cabang. Jika Anda tidak menentukan nilai, itu default ke wildcard (). *
6. (Opsional) Untuk Kondisi (opsional), pilih Tambahkan Kondisi untuk membuat kondisi tambahan yang harus dipenuhi sebelum pengguna aplikasi Anda dapat menggunakan izin yang diberikan peran. Misalnya, Anda dapat menambahkan kondisi yang memberikan akses ke AWS sumber

daya hanya untuk ID pengguna IAM tertentu. Anda juga dapat menambahkan ketentuan ke kebijakan kepercayaan setelah peran dibuat. Untuk informasi selengkapnya, lihat [Mengubah kebijakan kepercayaan peran \(konsol\)](#).

7. Tinjau informasi OIDC Anda dan kemudian 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 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 diri Anda Akun AWS. Mereka tidak bergantung pada 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 membuatnya.
12. (Opsional) Untuk Deskripsi, masukkan deskripsi untuk peran baru ini.
13. Untuk mengedit kasus penggunaan dan izin untuk peran, pilih Edit di Langkah 1: Pilih entitas tepercaya atau Langkah 2: Tambahkan izin bagian.
14. (Opsional) Untuk menambahkan metadata ke peran, lampirkan tag sebagai pasangan kunci-nilai. Untuk informasi selengkapnya tentang menggunakan tanda di IAM, lihat [Menandai sumber daya IAM](#).
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 ke GitHub organisasi, repositori, atau cabang tertentu. Anda dapat menggunakan kunci kondisi `token.actions.githubusercontent.com:sub` dengan operator kondisi string untuk

membatasi akses. Kami menyarankan Anda membatasi kondisi ke kumpulan repositori atau cabang tertentu dalam organisasi Anda GitHub . Untuk informasi tentang cara AWS mengonfigurasi OIDC trust GitHub sebagai identitas federasi, lihat [GitHub Dokumen - Mengonfigurasi OpenID Connect di Amazon Web Services](#).

Jika Anda menggunakan GitHub lingkungan dalam alur kerja tindakan atau dalam kebijakan OIDC, kami sangat menyarankan untuk menambahkan aturan perlindungan ke lingkungan untuk keamanan tambahan. Gunakan cabang dan tag penyebaran untuk membatasi cabang dan tag mana yang dapat diterapkan ke lingkungan. Untuk informasi selengkapnya tentang mengonfigurasi lingkungan dengan aturan perlindungan, lihat [Cabang dan tag penerapan](#) di GitHub artikel Menggunakan lingkungan untuk penerapan.

Kapan GitHub OIDC 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 bahwa nilainya bukan semata-mata karakter wildcard (* dan?) atau null. IAM melakukan pemeriksaan ini ketika 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, maka GitHub Tindakan dari organisasi atau repositori di luar kendali Anda dapat mengambil peran yang terkait dengan GitHub IDP 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.

```
{
  "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"
    }
  }
}

```

Contoh kondisi berikut membatasi akses ke GitHub organisasi dan repositori yang ditentukan, tetapi memberikan akses ke cabang mana pun dalam repositori.

```

"Condition": {
  "StringEquals": {
    "token.actions.githubusercontent.com:aud": "sts.amazonaws.com"
  },
  "StringLike": {
    "token.actions.githubusercontent.com:sub": "repo:GitHubOrg/GitHubRepo:*"
  }
}

```

Contoh kondisi berikut membatasi akses ke repositori atau cabang apa pun dalam organisasi yang ditentukan GitHub . Kami menyarankan Anda membatasi kunci kondisi `token.actions.githubusercontent.com:sub` ke nilai tertentu yang membatasi akses ke GitHub Tindakan 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, lihat [Kunci yang tersedia untuk AWS federasi OIDC](#)

Buat peran untuk federasi SAMP 2.0 (konsol)

Anda dapat menggunakan federasi SAMP 2.0 alih-alih membuat pengguna IAM di file 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 dan federasi identitas](#).

 Note

Untuk meningkatkan ketahanan federasi, kami menyarankan Anda mengonfigurasi IDP dan AWS federasi Anda untuk mendukung beberapa titik akhir masuk SAMP. Untuk detailnya, lihat artikel Blog AWS Keamanan [Cara menggunakan endpoint SAMP regional untuk failover](#).

Prasyarat untuk membuat peran untuk SAML

Sebelum Anda dapat membuat peran untuk federasi SAMP 2.0, Anda harus terlebih dahulu menyelesaikan langkah-langkah prasyarat berikut.

Untuk mempersiapkan pembuatan peran federasi SAML 2.0

1. Sebelum Anda membuat peran untuk federasi berbasis SAML, Anda harus membuat penyedia SAML di IAM. Untuk informasi selengkapnya, lihat [Buat penyedia identitas SAMP di IAM](#).
2. Siapkan 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 pengguna federasi.

Ketika Anda membuat kebijakan kepercayaan untuk peran Anda, Anda harus menggunakan tiga nilai untuk memastikan bahwa hanya aplikasi Anda yang dapat mengambil peran:

- 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 elemen `Condition`, gunakan syarat `StringEquals` untuk menguji bahwa atribut `saml:aud` dari respons SAML sesuai dengan titik akhir federasi SAML untuk AWS.

Contoh kebijakan kepercayaan berikut ini dirancang untuk pengguna federasi SAML:

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "sts:AssumeRoleWithSAML",
    "Principal": {"Federated": "arn:aws:iam::account-id:saml-provider/PROVIDER-NAME"},
    "Condition": {"StringEquals": {"SAML:aud": "https://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 persyaratan, Anda dapat membuat peran untuk federasi berbasis SAML.

Untuk membuat peran untuk federasi berbasis SAML

1. Masuk ke AWS Management Console 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 SAMP, 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 CLI
- Pilih Izinkan programatik dan AWS Management Console akses untuk membuat peran yang dapat diasumsikan secara terprogram dan dari. AWS Management Console

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 SAMP (SAML : audatribut) disetel ke titik akhir AWS masuk untuk SAMP (<https://signin.aws.amazon.com/saml>).

6. 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 akses terprogram dan konsol, SAML : aud atribut secara otomatis ditambahkan dan disetel ke URL titik akhir AWS SAMP (<https://signin.aws.amazon.com/saml>).

7. Untuk menambahkan lebih banyak kondisi terkait atribut ke kebijakan trust, 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 SAMP 2.0 Anda dan kemudian pilih Berikutnya.
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 Anda inginkan untuk dimiliki oleh pengguna federasi 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.

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: Peninjauan.

13. Untuk Nama peran, masukkan 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.

14. (Opsional) Untuk Deskripsi, masukkan deskripsi untuk peran baru ini.

15. Pilih Edit di Langkah 1: Pilih entitas tepercaya atau Langkah 2: Tambahkan bagian izin 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 [Menandai sumber daya IAM](#).

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 Anda inginkan agar digunakan oleh pengguna federasi Anda. Ini disebut sebagai mengonfigurasi kepercayaan pihak pengandal antara Idp dan AWS. Untuk informasi selengkapnya, lihat [Konfigurasi IDP SAMP 2.0 Anda dengan mengandalkan kepercayaan pihak dan menambahkan klaim](#).

Membuat peran menggunakan kebijakan kepercayaan khusus (konsol)

Anda dapat membuat kebijakan kepercayaan khusus untuk mendelegasikan akses dan memungkinkan orang lain untuk melakukan tindakan di AndaAkun 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 (konsole)

Anda dapat menggunakan peran AWS Management Console yang dapat diasumsikan oleh pengguna IAM. Misalnya, anggaplah bahwa organisasi Anda memiliki beberapa Akun AWS untuk mengisolasi lingkungan pengembangan dari lingkungan produksi. Untuk informasi tingkat tinggi tentang membuat peran yang memperbolehkan pengguna dalam akun pengembangan untuk mengakses sumber daya di akun produksi, lihat [Contoh skenario menggunakan akun pengembangan dan produksi terpisah](#).

Untuk membuat peran menggunakan kebijakan kepercayaan khusus (konsole)

1. Masuk ke AWS Management Console 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 khusus.
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. Centang di samping kebijakan kepercayaan khusus yang Anda buat.
7. (Opsional) Tetapkan [batas izin](#). Ini adalah fitur lanjutan yang tersedia untuk peran layanan, tetapi bukan peran tertaut layanan.

Buka bagian izin dan pilih Gunakan batas izin untuk mengontrol izin peran maksimum. IAM mencakup daftar kebijakan yang AWS dikelola dan pelanggan di akun Anda. Pilih kebijakan yang akan digunakan untuk batas izin.

8. Pilih Selanjutnya.
9. 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 dalam nama Akun AWS. Grup tidak dibedakan berdasarkan huruf besar-kecil. Misalnya, Anda tidak dapat membuat peran dengan nama **PRODRROLE** dan **prodrole**. Karena sumber

daya AWS lainnya mungkin merujuk peran tersebut, Anda tidak dapat mengubah nama peran tersebut setelah nama dibuat.

10. (Opsional) Untuk Deskripsi, masukkan deskripsi untuk peran baru tersebut.
11. Pilih Edit di Langkah 1: Pilih entitas tepercaya atau Langkah 2: Tambahkan bagian izin untuk mengedit kebijakan kustom dan izin untuk peran tersebut.
12. (Opsional) Tambahkan metadata ke peran dengan melampirkan tag sebagai pasangan nilai kunci. Untuk informasi selengkapnya tentang menggunakan tag di IAM, lihat [Menandai sumber daya IAM](#).
13. Tinjau peran dan kemudian pilih Buat peran.

Contoh kebijakan untuk mendelegasikan akses

Contoh-contoh berikut menunjukkan bagaimana Anda dapat mengizinkan atau memberikan Akun AWS akses ke sumber daya di 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 sumber daya sumber daya sumber daya sumber daya Akun AWS sumber daya lain](#)
- [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 sumber daya sumber daya sumber daya sumber daya Akun AWS sumber daya lain

Untuk tutorial yang menunjukkan cara menggunakan peran IAM untuk memberikan akses kepada pengguna dalam satu akun ke AWS sumber daya yang ada di akun lain, lihat [Tutorial IAM: Delegasikan akses 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 prinsipal 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 prinsipal muncul di konsol karena AWS tidak 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 mengasumsikan 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.

```
{
  "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 mybucket, dan nomor akun akun B adalah 111122223333. Itu tidak menentukan pengguna individual atau pengguna kelompok di akun B, hanya akun itu sendiri.

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Sid": "AccountBAccess1",  
      "Effect": "Allow",  
      "Principal": {"AWS": "111122223333"},  
      "Action": "s3:*",  
      "Resource": [  
        "arn:aws:s3:::mybucket",  
        "arn:aws:s3:::mybucket/*"  
      ]  
    }  
  ]  
}
```

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 mengontrol akses pada bucket S3 dan objek, kunjungi [Kontrol Akses](#) dalam Panduan Pengguna Amazon Simple Storage Service.

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.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:List*",
    "Resource": [
      "arn:aws:s3:::mybucket",
      "arn:aws:s3:::mybucket/*"
    ]
  }
}
```

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.

```
{
  "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).

```
{
  "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 antrian akun A. Namun, akun B hanya dapat mendelegasikan akses jika akun B telah diberi akses. Kelompok akun B yang memiliki kebijakan kedua dapat mengakses antrian hanya antara tengah hari hingga pukul 15.00 pada 30 November 2014. Pengguna hanya dapat melakukan tindakan SendMessage dan ReceiveMessage, sebagaimana ditentukan dalam kebijakan antrian Amazon SQS akun A.

Tidak dapat mendelegasikan akun ketika akun ditolak aksesnya

Akun AWS tidak dapat mendelegasikan akses ke sumber daya akun lain jika akun lain secara eksplisit telah 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 mybucket, dan nomor akun akun B adalah 1111-2222-3333. Akun A menggunakan Amazon S3 untuk menerapkan kebijakan ini.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "AccountBDeny",
    "Effect": "Deny",
    "Principal": {"AWS": "111122223333"},
    "Action": "s3:*",
    "Resource": "arn:aws:s3:::mybucket/*"
  }
}
```

Penolakan secara eksplisit ini membatalkan kebijakan apa pun dalam akun B yang memberikan izin untuk mengakses bucket S3 dalam akun A.

Menggunakan 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 AWS Management Console, Alat untuk Windows PowerShell, AWS Command Line Interface (AWS CLI) dan [AssumeRoleAPI](#).

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 AWS Management Console, maka gabungan Path dan RoleName tidak dapat melebihi 64 karakter.

Anda dapat beralih peran dari AWS Management Console. Anda dapat mengambil peran dengan memanggil operasi AWS CLI atau API atau dengan menggunakan URL kustom. Metode yang Anda gunakan menentukan siapa yang dapat mengasumsikan peran tersebut dan berapa lama sesi peran tersebut 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.

Membandingkan metode untuk menggunakan peran

Metode untuk mengasumsikan peran	Siapa yang bisa mengambil peran	Metode untuk menentukan masa pakai kredensi	Masa pakai kredensi (min maks default)
AWS Management Console	Pengguna (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 <code>duration-seconds</code> CLI atau <code>DurationSeconds</code> 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 <code>duration-seconds</code> CLI atau <code>DurationSeconds</code> API	15 menit Pengaturan durasi sesi maksimum ² 1 jam
Operasi assume-role-with-web-identity	Setiap pengguna yang diautentikasi menggunakan penyedia OIDC	Parameter <code>duration-seconds</code> CLI	15 menit Pengaturan durasi sesi

Metode untuk mengasumsikan peran	Siapa yang bisa mengambil peran	Metode untuk menentukan masa pakai kredensi	Masa pakai kredensi (min maks default)
ty CLI atau AssumeRoleWithWebIdentity API		atau <code>DurationSeconds API</code>	maksimum ² 1 jam
URL Konsol yang dibangun dengan <code>AssumeRole</code>	Pengguna atau peran	<code>SessionDuration</code> Parameter HTML dalam URL	15 menit 12 jam 1 jam
URL Konsol yang dibangun dengan <code>AssumeRoleWithSAML</code>	Pengguna yang diotentikasi menggunakan SAML	<code>SessionDuration</code> Parameter HTML dalam URL	15 menit 12 jam 1 jam
URL Konsol yang dibangun dengan <code>AssumeRoleWithWebIdentity</code>	Setiap pengguna yang diautentikasi menggunakan penyedia OIDC	<code>SessionDuration</code> Parameter HTML dalam URL	15 menit 12 jam 1 jam

¹ Menggunakan kredensial untuk satu peran untuk mengasumsikan peran 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.

² Pengaturan ini dapat memiliki nilai dari 1 jam hingga 12 jam. Untuk detail tentang perubahan pengaturan durasi sesi maksimum, lihat [Mengubah peran](#). 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. Anggaphlah 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 [Lihat pengaturan 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 kredensial ini. Ini memastikan bahwa Anda selalu memiliki seperangkat kredensial 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. [Mengubah kebijakan kepercayaan peran \(konsol\)](#)

Topik

- [Lihat pengaturan durasi sesi maksimum untuk peran](#)
- [Memberikan izin pengguna untuk berganti peran](#)
- [Memberi izin kepada pengguna untuk meneruskan peran ke layanan AWS](#)
- [Beralih ke peran \(konsol\)](#)
- [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 pada aplikasi yang berjalan di instans Amazon EC2.](#)
- [Mencabut kredensial keamanan sementara peran IAM](#)

Lihat pengaturan durasi sesi maksimum untuk peran

Anda dapat menentukan durasi sesi maksimum untuk peran menggunakan AWS Management Console atau dengan menggunakan AWS API AWS CLI atau. Saat Anda menggunakan operasi AWS CLI atau API untuk mengambil peran, Anda dapat menentukan nilai untuk `DurationSeconds` parameter tersebut. Gunakan parameter untuk menentukan durasi sesi peran, mulai dari 900 detik (15 menit) hingga pengaturan durasi sesi maksimum untuk peran tersebut. Sebelum menetapkan parameter, Anda harus melihat pengaturan ini untuk peran Anda. Jika Anda menentukan nilai untuk parameter `DurationSeconds` yang lebih tinggi dari pengaturan maksimum, operasi gagal.

Untuk melihat durasi sesi maksimum peran (konsol)

1. Di panel navigasi konsol IAM, pilih Peran.
2. Pilih nama peran yang ingin Anda lihat.
3. Di samping Durasi sesi maksimum, lihat panjang sesi maksimum yang diberikan untuk peran tersebut. Ini adalah durasi sesi maksimum yang dapat Anda tentukan dalam operasi Anda AWS CLI, atau API.

Untuk melihat pengaturan durasi sesi maksimum peran (AWS CLI)

1. Jika Anda tidak tahu nama peran yang ingin Anda asumsikan, jalankan perintah berikut untuk membuat daftar peran di akun Anda:
 - [aws iam daftar peran](#)
2. Untuk melihat durasi sesi maksimum peran, jalankan perintah berikut. Lalu lihat parameter durasi sesi maksimum.
 - [aws iam mendapatkan peran](#)

Untuk melihat setelan durasi sesi maksimum (AWS API) peran

1. Jika Anda tidak tahu nama peran yang ingin Anda jalankan, lakukan operasi berikut untuk membuat daftar peran di akun Anda:

- [ListRoles](#)
2. Untuk melihat durasi sesi maksimum peran, lakukan operasi berikut. Lalu lihat parameter durasi sesi maksimum.
 - [GetRole](#)

Memberikan izin pengguna untuk berganti peran

Saat administrator [membuat peran untuk akses lintas akun](#) mereka membangun kepercayaan antara akun yang memiliki peran tersebut, sumber daya (akun terpercaya), dan akun yang berisi pengguna (rekening terpercaya). Untuk melakukannya, administrator akun terpercaya menentukan nomor akun terpercaya sebagai `Principal` dalam kebijakan kepercayaan peran tersebut. Hal ini berpotensi memungkinkan semua pengguna dalam akun terpercaya untuk mengasumsikan 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 terpercaya, buat kebijakan baru untuk pengguna tersebut, atau ubah kebijakan yang ada untuk menambahkan elemen yang diperlukan. Untuk detailnya, lihat [Membuat atau mengubah](#).
2. Kemudian, pilih cara Anda ingin berbagi informasi peran:
 - Tautan peran: Kirim tautan kepada pengguna yang membawa mereka ke halaman Ganti Peran dengan detail yang sudah diisi.
 - ID akun atau alias: Berikan nama peran bersama dengan nomor ID akun atau alias akun. Pengguna kemudian masuk ke halaman Ganti Peran dan menambahkan detail secara manual.

Untuk detailnya, lihat [Memberikan informasi kepada pengguna](#).

Perhatikan bahwa Anda dapat berganti peran hanya saat Anda masuk sebagai pengguna IAM, peran federasi SAML-, atau peran federasi identitas web. Anda tidak dapat beralih peran saat masuk sebagai Pengguna root akun AWS.

⚠ Important

Anda tidak dapat berganti peran AWS Management Console ke peran yang memerlukan [ExternalId](#) nilai. Anda hanya dapat beralih ke peran tersebut dengan memanggil [AssumeRole](#) API yang mendukung parameter `ExternalId`.

ℹ Catatan

- Topik ini membahas kebijakan untuk pengguna, karena Anda pada akhirnya memberikan izin kepada pengguna untuk menyelesaikan tugas. Namun, kami tidak menyarankan agar Anda memberikan izin langsung kepada pengguna individual. Ketika pengguna mengasumsikan peran, mereka diberi izin yang terkait dengan peran tersebut.
- Saat Anda beralih peran di AWS Management Console, konsol selalu menggunakan kredensial asli Anda untuk mengotorisasi peralihan. Hal ini berlaku baik Anda masuk sebagai pengguna IAM, sebagai peran federasi SAML, atau sebagai peran federasi identitas-web. Misalnya, jika Anda beralih ke RoleA, IAM menggunakan kredensi pengguna asli atau peran federasi untuk menentukan apakah Anda diizinkan untuk mengambil 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 berganti peran pada sumber daya yang terdaftar (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.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::account-id: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 menyederhanakan akses untuk pengguna dengan mengirim tautan yang telah dikonfigurasi sebelumnya dengan ID akun dan nama peran. Anda dapat melihat tautan peran setelah menyelesaikan wizard Buat Peran dengan memilih banner Lihat Peran, atau pada halaman Ringkasan Peran untuk setiap peran yang diaktifkan dengan 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 sarankan agar Anda mengarahkan pengguna Anda ke [Beralih ke peran \(konsol\)](#) untuk memandu mereka melalui proses tersebut. Untuk memecahkan masalah umum yang mungkin Anda temui saat mengasumsikan peran, lihat [Saya tidak dapat mengsumsikan peran](#).

Pertimbangan-pertimbangan

- Apabila Anda membuat program peran secara terprogram, Anda juga 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 AWS Management Console. Sebagai contoh: `division_abc/subdivision_efg/role_XYZ`.
- Apabila Anda membuat program peran secara terprogram, Anda juga dapat menambahkan Path hingga 512 karakter dan. RoleName Panjang nama peran dapat mencapai 64 karakter. Namun, untuk menggunakan peran dengan fitur Ganti Peran di AWS Management Console, Path dan RoleName yang sudah digabungkan tidak boleh lebih dari 64 karakter.
- Untuk tujuan keamanan, Anda dapat [meninjau log AWS CloudTrail](#) untuk mempelajari siapa yang melakukan tindakan di AWS. Anda dapat menggunakan kunci syarat `sts:SourceIdentity` dalam peran kebijakan kepercayaan untuk mengharuskan pengguna menentukan identitas saat mereka mengasumsikan sebuah peran. Misalnya, Anda dapat meminta pengguna IAM

untuk 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.

Memberi izin kepada pengguna untuk meneruskan peran ke layanan AWS

Untuk mengonfigurasi banyak AWS layanan, Anda harus meneruskan 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 instance yang menyediakan kredensi 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 layanan AWS, 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 di 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 dalam Akun B dapat diteruskan ke layanan. Untuk detailnya, lihat [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 menyetel `PassRole` 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 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, Auto Scaling Amazon EC2 membuat peran terkait layanan untuk Anda `AWSServiceRoleForAutoScaling` saat membuat grup Auto Scaling untuk pertama kalinya. Jika Anda mencoba menentukan peran terkait layanan saat membuat grup Auto Scaling dan Anda tidak memiliki `iam:PassRole` izin, Anda akan menerima kesalahan. Jika Anda tidak menentukan peran secara eksplisit, `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: Condition](#). Kunci syarat `iam:PassedToService` dapat digunakan untuk menentukan ke prinsipal layanan mana sebuah peran dapat diberikan. Untuk mempelajari selengkapnya tentang menggunakan kunci `iam:PassedToService` kondisi dalam kebijakan, lihat [iam: PassedToService](#).

Contoh 1

Misalkan Anda ingin memberi pengguna kemampuan untuk meneruskan serangkaian peran yang disetujui ke layanan Amazon EC2 setelah meluncurkan instance. Anda memerlukan tiga elemen:

- Kebijakan izin IAM yang dilampirkan 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.

```
{
  "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.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "TrustPolicyStatementThatAllowsEC2ServiceToAssumeTheAttachedRole",
    "Effect": "Allow",
    "Principal": { "Service": "ec2.amazonaws.com" },
    "Action": "sts:AssumeRole"
  }
}
```

- Kebijakan izin IAM yang dilampirkan pada pengguna IAM yang memungkinkan pengguna untuk hanya meneruskan peran yang disetujui tersebut. Anda biasanya `iam:GetRole` menambahkan `iam:PassRole` sehingga pengguna bisa mendapatkan detail peran yang akan diteruskan. Dalam contoh ini, pengguna hanya dapat meneruskan peran yang ada di akun yang ditentukan dengan nama yang diawali dengan `EC2-roles-for-XYZ-`:

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

```
    "Effect": "Allow",
    "Action": [
        "iam:GetRole",
        "iam:PassRole"
    ],
    "Resource": "arn:aws:iam::account-id: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 dilampirkan 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 AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Pilih Peran, lalu pilih Buat peran.
3. Pilih jenis peran AWSLayanan, dan kemudian untuk kasus Penggunaan untuk lainnya Layanan AWS, pilih layanan RDS. Pilih RDS — Enhanced Monitoring, lalu pilih Next.
4. Pilih kebijakan izin AmazonRDS. EnhancedMonitoringRole
5. Pilih Berikutnya.
6. Untuk nama Peran, masukkan nama peran yang membantu Anda mengidentifikasi tujuan peran ini. Nama peran harus unik di dalam diri AndaAkun 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.

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 [Menandai sumber daya IAM](#).
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 Enhanced Monitoring 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 di 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 RDS-, Anda dapat mengganti nama peran di ARN sumber daya dengan wildcard, sebagai berikut.

```
"Resource": "arn:aws:iam::account-id:role/RDS-*
```

iam:PassRole tindakan dalam AWS CloudTrail log

PassRole bukan panggilan API. PassRole adalah izin, artinya tidak ada CloudTrail log yang dihasilkan untuk IAMPassRole. Untuk meninjau peran apa yang diteruskan ke mana Layanan AWS CloudTrail, Anda harus meninjau CloudTrail log yang membuat atau memodifikasi AWS sumber 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.

Beralih ke peran (konsol)

Peran menentukan serangkaian izin yang dapat Anda gunakan untuk mengakses sumber daya AWS yang Anda perlukan. Dalam hal ini, 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 [Membuat Peran IAM](#).

Important

Izin pengguna Anda dan peran apa pun yang Anda ambil 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 beralih peran di AWS Management Console, konsol selalu menggunakan kredensial asli Anda untuk mengotorisasi peralihan. Ini berlaku baik Anda masuk sebagai pengguna IAM, pengguna di Pusat Identitas IAM, sebagai peran federasi SAML, atau sebagai peran federasi identitas web. Misalnya, jika Anda beralih ke RoleA, IAM menggunakan pengguna asli atau kredensi peran federasi untuk menentukan apakah Anda diizinkan untuk mengasumsikan RoleA. Jika Anda kemudian beralih ke RoleB saat menggunakan RoleA, AWS masih menggunakan pengguna asli atau kredensial peran gabungan untuk mengotorisasi peralihan, bukan kredensial untuk RoleA.

Hal-hal yang perlu diketahui tentang pergantian peran dalam konsol

Bagian ini memberikan informasi tambahan tentang penggunaan konsol IAM untuk beralih ke peran.

i Catatan:

- Anda tidak dapat beralih peran jika masuk sebagai Pengguna root akun AWS. Anda dapat beralih peran saat masuk sebagai pengguna IAM, pengguna di Pusat Identitas IAM, peran federasi SAML, atau peran federasi identitas web.
- Anda tidak dapat beralih peran AWS Management Console ke peran yang membutuhkan [ExternalId](#) nilai. Anda hanya dapat beralih ke peran tersebut dengan memanggil [AssumeRole](#) API yang mendukung parameter `ExternalId`.

- Jika administrator memberi Anda tautan, pilih tautan lalu lewati ke langkah [Step 5](#) dalam prosedur berikut. Tautan tersebut membawa Anda ke halaman web yang sesuai dan mengisi ID akun (atau alias) dan nama peran.
- Anda dapat membuat tautan secara manual lalu melompat ke langkah [Step 5](#) dalam prosedur berikut. Untuk membuat tautan Anda, gunakan format berikut:

```
https://signin.aws.amazon.com/switchrole?  
account=account_id_number&roleName=role_name&displayName=text_to_display
```

Jika Anda mengganti teks berikut:

- *account_id_number* — Pengenal akun 12 digit yang diberikan kepada Anda oleh administrator Anda. Atau, administrator Anda dapat membuat alias akun sehingga URL menyertakan nama akun Anda alih-alih ID akun. Untuk informasi selengkapnya, lihat [Jenis Pengguna](#) di Panduan AWS Sign-In Pengguna.
- *role_name* — Nama peran yang ingin Anda asumsikan. Anda bisa mendapatkan ini dari akhir ARN peran. Misalnya, berikan `TestRole` nama peran dari ARN peran berikut:
`arn:aws:iam::123456789012:role/TestRole`.
- (Opsional) *text_to_display* — Teks yang ingin Anda tampilkan di bilah navigasi menggantikan nama pengguna Anda saat peran ini aktif.
- Anda dapat berganti peran secara manual dengan menggunakan informasi yang disediakan administrator Anda dengan menggunakan prosedur berikut.

Secara default, saat Anda beralih peran, AWS Management Console sesi Anda berlangsung selama 1 jam. 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. Misalnya, anggaplah durasi sesi maksimum 10 jam ditetapkan untuk peran. Pengguna IAM telah masuk ke konsol selama 8 jam saat mereka memutuskan untuk beralih peran tersebut. Ada sisa waktu 4 jam di sesi pengguna, jadi durasi sesi peran yang diizinkan adalah 4 jam. Tabel berikut menunjukkan cara menentukan durasi sesi untuk pengguna IAM saat berganti peran di dalam konsol.

Durasi sesi peran konsol pengguna IAM

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)		

Note

Beberapa konsol layanan AWS dapat memperbarui sesi peran secara otomatis saat sesi berakhir tanpa Anda harus melakukan tindakan apa pun. Beberapa mungkin akan meminta Anda memuat ulang halaman peramban untuk mengotentikasi ulang sesi Anda.

Untuk memecahkan masalah umum yang mungkin Anda temui saat mengasumsikan peran, lihat [Saya tidak dapat mengasumsikan peran](#).

Untuk beralih ke peran (konsol)

1. [Masuk ke AWS Management Console sebagai pengguna IAM dan buka konsol IAM di https://console.aws.amazon.com/iam/.](https://console.aws.amazon.com/iam/)

2. Di konsol IAM, pilih nama pengguna Anda di bilah navigasi di kanan atas. Terlihat seperti ini:
username@account_ID_number_or_alias.
3. Pilih Ganti Peran. Jika ini pertama kalinya Anda memilih opsi ini, akan muncul halaman dengan informasi lebih lanjut. Setelah itu, pilih Ganti Peran. Jika Anda menghapus cookie peramban Anda, halaman ini dapat muncul lagi.
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 menyetik jalur dan nama lengkap tersebut di kotak Peran. Jika Anda hanya menyetik 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) Pilih Nama Tampilan. Ketikkan teks yang ingin Anda tampilkan di bilah navigasi di tempat 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. Anda juga dapat memilih 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.

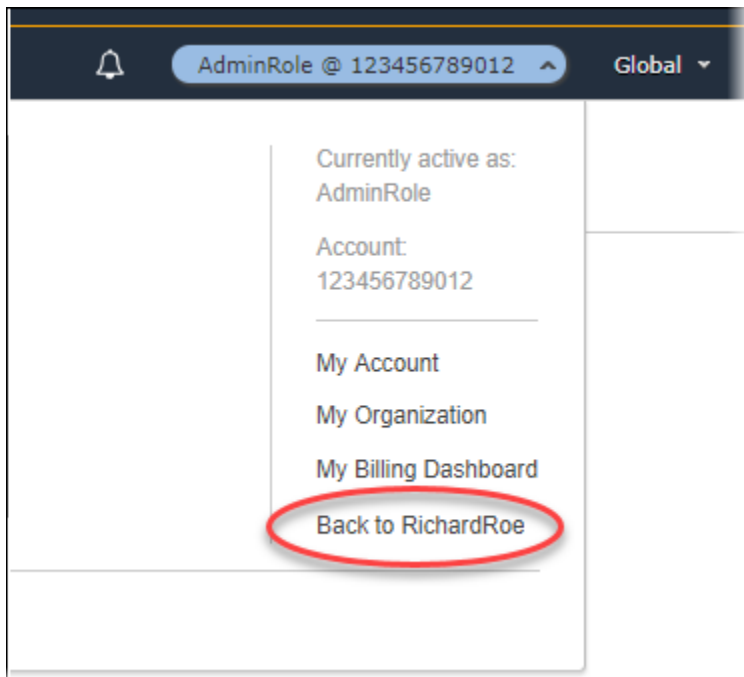
Kiat

Beberapa peran terakhir yang Anda gunakan muncul pada menu. Kali berikutnya Anda perlu beralih ke salah satu peran tersebut, Anda cukup memilih peran yang Anda inginkan. Anda hanya perlu menyetikkan informasi akun dan peran secara manual jika peran tidak ditampilkan pada menu.

Untuk berhenti menggunakan peran (konsol)

1. Di konsol IAM, pilih Nama tampilan peran Anda di bilah navigasi di kanan atas. Terlihat seperti ini: ***rolename@account_ID_number_or_alias***.
2. Pilih Kembali ke ***nama pengguna***. Peran dan izinnya dinonaktifkan, dan izin yang terkait dengan pengguna dan grup IAM Anda dipulihkan secara otomatis.

Misalnya, anggap Anda masuk ke nomor akun 123456789012 menggunakan nama pengguna RichardRoe. Setelah menggunakan peran AdminRole, Anda ingin berhenti menggunakan peran dan kembali ke izin awal. Untuk berhenti menggunakan peran, pilih AdminRole @ 123456789012, lalu pilih Kembali ke RichardRoe.



Beralih ke peran IAM (AWS CLI)

Peran menentukan serangkaian izin yang dapat Anda gunakan untuk mengakses sumber daya AWS yang Anda perlukan. Dalam hal itu, ini mirip dengan [pengguna dalam 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. Peran ini dapat ditempatkan di akun Anda sendiri atau lainnya Akun AWS. Untuk informasi lebih lanjut tentang peran, keuntungannya, dan

cara membuatnya, lihat [Peran IAM](#), dan [Membuat Peran IAM](#). Untuk mempelajari tentang berbagai metode yang dapat Anda gunakan untuk mengasumsikan peran, lihat [Menggunakan 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 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 masuk sebagai pengguna IAM. Anda juga dapat menggunakan peran untuk menjalankan perintah AWS CLI saat masuk sebagai [pengguna yang diotentikasi secara eksternal](#) ([SAML](#) atau [OIDC](#)) yang telah 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.

[Rantai peran](#) - Anda juga dapat menggunakan rantai peran, yaitu 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 beralih peran di konsol, kredenik baru dibatasi hingga durasi maksimum satu jam. Untuk mempelajari cara melihat nilai maksimum untuk peran Anda, lihat [Lihat pengaturan 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

pengguna federasi, itu dapat menjadi key pair akses untuk peran yang awalnya ditetapkan kepada Anda. Jika izin Anda saat ini memberi Anda kemampuan untuk mengasumsikan peran IAM tertentu, maka Anda dapat mengidentifikasi peran tersebut dalam file AWS CLI konfigurasi. Perintah tersebut kemudian dijalankan atas izin peran IAM yang ditentukan, bukan identitas awal. Perhatikan bahwa ketika Anda menentukan profil tersebut dalam perintah AWS CLI, Anda menggunakan peran baru tersebut. 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 log AWS CloudTrail](#) untuk mempelajari 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 belum pernah menggunakan AWS CLI, Anda harus terlebih dahulu mengonfigurasi profil CLI default. Buka command prompt dan atur AWS CLI instalasi Anda untuk menggunakan kunci akses dari pengguna IAM 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. Saat profil ini diaktifkan, AWS CLI menggunakan kredensial `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, setiap AWS CLI perintah yang menentukan parameter `--profile prodaccess` berjalan berdasarkan izin yang dilampirkan ke peran `IAMProductionAccessRole` alih-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`. AWS CLI kembali menggunakan kredensial dalam profil default Anda, yang telah dikonfigurasi dalam [Step 1](#).

Untuk informasi selengkapnya, lihat [Mengasumsikan Peran](#) di Panduan AWS Command Line Interface Pengguna.

Skenario contoh: Mengizinkan 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 tersebut. Anggap bahwa instans EC2 ada di akun 111111111111. Instans tersebut menyertakan peran profil `abcd` instans yang memungkinkan aplikasi melakukan tugas Amazon S3 hanya-baca `dimy-bucket-1` bucket dalam 111111111111 akun yang sama. 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 `abcd` peran akun 111111111111

```
{
  "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:::my-bucket-1/*",
      "arn:aws:s3:::my-bucket-1"
    ]
  },
  {
    "Sid": "AllowIPToAssumeCrossAccountRole",
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::222222222222:role/efgh"
  }
]
}

```

Anggap bahwa `efgh` peran lintas akun memungkinkan tugas Amazon S3 hanya baca di bucket `my-bucket-2` dengan akun `222222222222` yang sama. Untuk melakukannya, peran lintas akun `efgh` harus memiliki kebijakan izin berikut:

Akun `222222222222` kebijakan izin **`efgh`** peran

```

{
  "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:::my-bucket-2/*",
            "arn:aws:s3:::my-bucket-2"
        ]
    }
]
}
}

```

Peran *efgh* harus memungkinkan peran profil instans *abcd* untuk mengasumsikannya. Untuk melakukannya, peran *efgh* harus memiliki kebijakan kepercayaan berikut:

Akun 222222222222 kebijakan kepercayaan ***efgh*** peran

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "efghTrustPolicy",
      "Effect": "Allow",
      "Action": "sts:AssumeRole",
      "Principal": {"AWS": "arn:aws:iam::111111111111:role/abcd"}
    }
  ]
}

```

Untuk kemudian menjalankan perintah AWS CLI dalam akun 222222222222, 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. Saat AWS CLI mengambil peran atas nama pengguna seperti yang dijelaskan dalam topik ini, nama sesi peran akan dibuat secara otomatis sebagai `AWS-CLI-session-nnnnnnnn`. Di sini, *nnnnnnnn* adalah bilangan bulat yang mewakili waktu dalam [jangka waktu Unix](#) (jumlah detik sejak tengah malam UTC pada 1 Januari 1970). Untuk informasi selengkapnya, lihat [ReferensiCloudTrail Peristiwa](#) di PanduanAWS 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, setiap perintah AWS CLI yang menentukan parameter `--profile instancecrossaccount` berjalan berdasarkan izin yang dilampirkan ke peran `efgh` dalam akun 222222222222.

```
aws s3 ls my-bucket-2 --profile instancecrossaccount
```

Perintah ini berfungsi jika izin yang diberikan ke `fg` peran mengizinkan pencantuman pengguna di 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 [Megasumsikan 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 hal 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. Peran ini dapat ditemukan di akun Anda sendiri atau lainnya Akun AWS. Untuk informasi lebih lanjut tentang peran, keuntungannya, dan cara membuatnya, lihat [Peran IAM](#), dan [Membuat 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 Anda kadang-kadang perlu bekerja dengan lingkungan produksi di baris perintah menggunakan [Alat 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 pengguna gabungan, itu dapat menjadi pasangan access key untuk peran yang awalnya ditetapkan kepada Anda. Anda dapat menggunakan kredensial ini untuk menjalankan `Use-STSRole` cmdlet

yang melewati ARN peran baru sebagai parameter. Perintah tersebut mengembalikan kredensial keamanan sementara 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 log AWS CloudTrail](#) untuk mempelajari 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 konfigurasi profil default untuk menggunakan access key dari pengguna IAM Anda saat ini atau peran federasi Anda. Jika sebelumnya Anda telah menggunakan Alat untuk Windows PowerShell, maka ini mungkin 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 [MenggunakanAWS Kredensi](#) di PanduanAWS Tools for Windows 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+qYf4nD00EXAMPLEEmj4wxS04L/uZEXAMPLECiHzFB51TYLto9dyBgSDyEXAMPLE9/
g7QRUhZp4bqbEXAMPLENwGPy
0j59pFA41NKCIkVgkREXAMPLEj1zxQ7y52gekeVEXAMPLEDiB9ST3UuysgsKdEXAMPLE1TVastU1A0SKFEXAMPLEiyw
C
s8EXAMPLEpZg0s+6hz4AP4KEXAMPLERbASP+4eZScEXAMPLEsnf87eNhyDHq6ikBQ==
```

```
PS C:\> $Creds.Expiration
Thursday, June 18, 2018 2:28:31 PM
```

- 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
```

- Untuk kembali ke kredensi asli Anda, cukup berhenti menggunakan `-Credentials $Creds` parameter dan memungkinkan PowerShell untuk kembali ke kredensial yang disimpan di profil default.

Beralih ke peran IAM (AWS API)

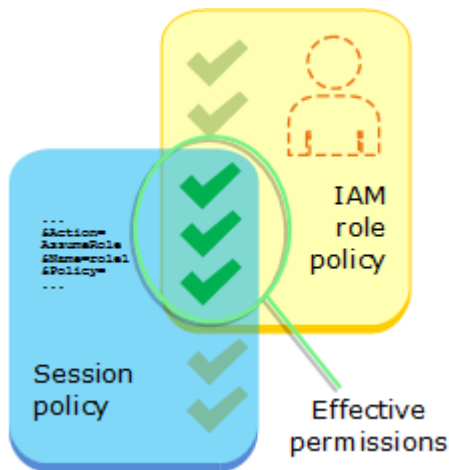
Peran menentukan serangkaian izin yang dapat Anda gunakan untuk mengakses sumber daya AWS. Dalam hal itu, ini mirip dengan [Pengguna IAM](#). Seorang kepala sekolah (orang atau aplikasi) mengambil peran untuk menerima izin sementara untuk melaksanakan tugas yang diperlukan dan berinteraksi dengan AWS 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 [Membuat Peran IAM](#). Untuk mempelajari tentang berbagai metode yang dapat Anda gunakan untuk mengasumsikan peran, lihat [Menggunakan 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 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 yang Anda teruskan sebagai parameter saat Anda secara membuat sesi sementara terprogram untuk peran atau pengguna federasi. 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. 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, lihat [Logika 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 [Lihat pengaturan 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 kredensial sementara, lalu bersihkan sumber daya.

.NET**AWS SDK for .NET****ℹ Note**

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya 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 IAMIdentityManagementService _IAMService;

/// <summary>
/// Constructor for the IAMWrapper class.
/// </summary>
/// <param name="IAMService">An IAM client object.</param>
public IAMWrapper(IAMIdentityManagementService IAMService)
{
    _IAMService = IAMService;
}

/// <summary>
/// Add an existing IAM user to an existing IAM group.
/// </summary>
/// <param name="userName">The username of the user to add.</param>
/// <param name="groupName">The name of the group to add the user to.</param>
/// <returns>A Boolean value indicating the success of the action.</returns>
public async Task<bool> AddUserToGroupAsync(string userName, string
groupName)
{
    var response = await _IAMService.AddUserToGroupAsync(new
AddUserToGroupRequest
    {
        GroupName = groupName,
        UserName = userName,
    });

    return response.HttpStatusCode == HttpStatusCode.OK;
}

/// <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 group.
/// </summary>
/// <param name="groupName">The name to give the IAM group.</param>
/// <returns>The IAM group that was created.</returns>
public async Task<Group> CreateGroupAsync(string groupName)
{
    var response = await _IAMService.CreateGroupAsync(new CreateGroupRequest
{ GroupName = groupName });
    return response.Group;
}

/// <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 group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteGroupAsync(string groupName)
    {
        var response = await _IAMService.DeleteGroupAsync(new DeleteGroupRequest
        { GroupName = groupName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// Delete an IAM policy associated with an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group associated with the
    /// policy.</param>
    /// <param name="policyName">The name of the policy to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteGroupPolicyAsync(string groupName, string
    policyName)
    {
        var request = new DeleteGroupPolicyRequest()
        {
            GroupName = groupName,
            PolicyName = policyName,
        };

        var response = await _IAMService.DeleteGroupPolicyAsync(request);
        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>
/// Remove a user from an IAM group.
/// </summary>
/// <param name="userName">The username of the user to remove.</param>
/// <param name="groupName">The name of the IAM group to remove the user
from.</param>
/// <returns>A Boolean value indicating the success of the action.</returns>
public async Task<bool> RemoveUserFromGroupAsync(string userName, string
groupName)
{
    // Remove the user from the group.
    var removeUserRequest = new RemoveUserFromGroupRequest()
    {
        UserName = userName,
        GroupName = groupName,
    };

    var response = await
_IAMService.RemoveUserFromGroupAsync(removeUserRequest);
    return response.HttpStatusCode == HttpStatusCode.OK;
}

/// <summary>
```

```
/// Add or update an inline policy document that is embedded in an IAM group.
/// </summary>
/// <param name="groupName">The name of the IAM group.</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> PutGroupPolicyAsync(string groupName, string
policyName, string policyDocument)
{
    var request = new PutGroupPolicyRequest
    {
        GroupName = groupName,
        PolicyName = policyName,
        PolicyDocument = policyDocument
    };

    var response = await _IAMService.PutGroupPolicyAsync(request);
    return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
}

/// <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();
}
}
```

- Lihat detail API di topik-topik berikut dalam Referensi API AWS SDK for .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
    fi
}
```

```

    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 getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function iam_create_user"
        echo "Creates an WS 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 getopt "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 getopt 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 getopt "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 getopt 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 getopt "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
}
```

```

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 getopt 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 getopt 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 getopt "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 role.
#
# 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 getopt 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 getopt "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 getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function iam_delete_policy"
        echo "Deletes an WS 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 getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function iam_delete_role"
        echo "Deletes an WS Identity and Access Management (IAM) role"
        echo "  -n role_name -- The name of the IAM role."
        echo ""
    }

    # Retrieve the calling parameters.
    while getopt "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 getopt command in a function.

```

```
# bashsupport disable=BP5008
function usage() {
    echo "function iam_delete_access_key"
    echo "Deletes an WS 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 getopt command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_user"
    echo "Deletes an WS 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 for C++

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya 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;
}
```

- Lihat detail API di topik-topik berikut dalam Referensi API AWS SDK for C++ .
 - [AttachRolePolicy](#)
 - [CreateAccessKey](#)
 - [CreatePolicy](#)
 - [CreateRole](#)
 - [CreateUser](#)
 - [DeleteAccessKey](#)
 - [DeletePolicy](#)
 - [DeleteRole](#)
 - [DeleteUser](#)
 - [DeleteUserPolicy](#)
 - [DetachRolePolicy](#)
 - [PutUserPolicy](#)

Go

SDK for Go V2

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di [Repositori Contoh Kode AWS](#).

Jalankan skenario interaktif di penggugah/prompt perintah.


```
// 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() {
    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()
    accessKey := scenario.CreateAccessKey(user)
    role := scenario.CreateRoleAndPolicies(user)
    noPermsConfig := scenario.ListBucketsWithoutPermissions(accessKey)
    scenario.ListBucketsWithAssumedRole(noPermsConfig, role)
    scenario.Cleanup(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() *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(userName)
    if err != nil {
        panic(err)
    }
}
```

```

}
if user == nil {
    user, err = scenario.userWrapper.CreateUser(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(user *types.User)
*types.AccessKey {
    accessKey, err := scenario.userWrapper.CreateAccessKeyPair(*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(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(scenario.helper.GetName(), *user.Arn)
    if err != nil {panic(err)}
    log.Printf("Created role %v.\n", *listBucketsRole.RoleName)
    listBucketsPolicy, err := scenario.policyWrapper.CreatePolicy(

```

```

    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(*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(*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(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(context.TODO(),
    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(context.TODO(), &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(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(context.TODO(),
    &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(context.TODO(),
    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(context.TODO(), &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(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(*role.RoleName)
        if err != nil {panic(err)}
    }
}

```

```

    for _, policy := range policies {
        err = scenario.roleWrapper.DetachRolePolicy(*role.RoleName,
*policy.PolicyArn)
        if err != nil {panic(err)}
        err = scenario.policyWrapper.DeletePolicy(*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(*role.RoleName)
    if err != nil {panic(err)}
    log.Printf("Deleted role %v.\n", *role.RoleName)

    userPols, err := scenario.userWrapper.ListUserPolicies(*user.UserName)
    if err != nil {panic(err)}
    for _, userPol := range userPols {
        err = scenario.userWrapper.DeleteUserPolicy(*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(*user.UserName)
    if err != nil {panic(err)}
    for _, key := range keys {
        err = scenario.userWrapper.DeleteAccessKey(*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(*user.UserName)
    if err != nil {panic(err)}
    log.Printf("Deleted user %v.\n", *user.UserName)
    log.Println(strings.Repeat("-", 88))
}
}

```

Tentukan struct yang membungkus tindakan akun.

```

// 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() (*types.PasswordPolicy,
    error) {
    var pwPolicy *types.PasswordPolicy
    result, err := wrapper.IamClient.GetAccountPasswordPolicy(context.TODO(),
        &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() ([]types.SAMLProviderListEntry,
    error) {
    var providers []types.SAMLProviderListEntry
    result, err := wrapper.IamClient.ListSAMLProviders(context.TODO(),
        &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.


```
// 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"`
}

// 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(maxPolicies int32) ([]types.Policy,
    error) {
    var policies []types.Policy
    result, err := wrapper.IamClient.ListPolicies(context.TODO(),
        &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
}
```

```
// 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(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(context.TODO(),
        &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(policyArn string) (*types.Policy, error) {
    var policy *types.Policy
    result, err := wrapper.IamClient.GetPolicy(context.TODO(), &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(policyArn string) error {
  _, err := wrapper.IamClient.DeletePolicy(context.TODO(), &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.

```

// 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(maxRoles int32) ([]types.Role, error) {
  var roles []types.Role
  result, err := wrapper.IamClient.ListRoles(context.TODO(),
    &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(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(context.TODO(),
        &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(roleName string) (*types.Role, error) {
    var role *types.Role
    result, err := wrapper.IamClient.GetRole(context.TODO(),
        &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(serviceName string,
    description string) (*types.Role, error) {
    var role *types.Role
    result, err := wrapper.IamClient.CreateServiceLinkedRole(context.TODO(),
        &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(roleName string) error {
    _, err := wrapper.IamClient.DeleteServiceLinkedRole(context.TODO(),
        &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(policyArn string, roleName string)
    error {
    _, err := wrapper.IamClient.AttachRolePolicy(context.TODO(),
        &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(roleName string)
    ([]types.AttachedPolicy, error) {
    var policies []types.AttachedPolicy
    result, err := wrapper.IamClient.ListAttachedRolePolicies(context.TODO(),
        &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(roleName string, policyArn string)
    error {
    _, err := wrapper.IamClient.DetachRolePolicy(context.TODO(),
    &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(roleName string) ([]string, error) {
    var policies []string
    result, err := wrapper.IamClient.ListRolePolicies(context.TODO(),
    &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(roleName string) error {
    _, err := wrapper.IamClient.DeleteRole(context.TODO(), &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.

```
// 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(maxUsers int32) ([]types.User, error) {
    var users []types.User
    result, err := wrapper.IamClient.ListUsers(context.TODO(), &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(userName string) (*types.User, error) {
    var user *types.User
    result, err := wrapper.IamClient.GetUser(context.TODO(), &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(userName string) (*types.User, error) {
    var user *types.User
    result, err := wrapper.IamClient.CreateUser(context.TODO(),
        &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(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(context.TODO(),
&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(userName string) ([]string, error) {
    var policies []string
    result, err := wrapper.IamClient.ListUserPolicies(context.TODO(),
&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(userName string, policyName string)
error {
    _, err := wrapper.IamClient.DeleteUserPolicy(context.TODO(),
    &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(userName string) error {
    _, err := wrapper.IamClient.DeleteUser(context.TODO(), &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(userName string)
(*types.AccessKey, error) {
    var key *types.AccessKey
    result, err := wrapper.IamClient.CreateAccessKey(context.TODO(),
    &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(userName string, keyId string) error {
_, err := wrapper.IamClient.DeleteAccessKey(context.TODO(),
&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(userName string)
([]types.AccessKeyMetadata, error) {
var keys []types.AccessKeyMetadata
result, err := wrapper.IamClient.ListAccessKeys(context.TODO(),
&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
}
```

- Lihat detail API di topik-topik berikut dalam Referensi API AWS SDK for Go .
 - [AttachRolePolicy](#)
 - [CreateAccessKey](#)
 - [CreatePolicy](#)
 - [CreateRole](#)
 - [CreateUser](#)
 - [DeleteAccessKey](#)
 - [DeletePolicy](#)
 - [DeleteRole](#)
 - [DeleteUser](#)
 - [DeleteUserPolicy](#)
 - [DetachRolePolicy](#)
 - [PutUserPolicy](#)

Java

SDK for 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);
    }
}
}
```

- Lihat detail API di topik-topik berikut dalam 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 kredensial sementara untuk membuat daftar bucket untuk akun.

```
import {
  CreateUserCommand,
  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";

// Set the parameters.
```

```
const iamClient = new IAMClient({});
const userName = "test_name";
const policyName = "test_policy";
const roleName = "test_role";

export const main = async () => {
  // Create a user. The user has no permissions by default.
  const { User } = await iamClient.send(
    new CreateUserCommand({ UserName: userName }),
  );

  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: 60 }, () =>
  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"));
};
```

- Lihat detail API di topik-topik berikut dalam Referensi API AWS SDK for JavaScript .
 - [AttachRolePolicy](#)
 - [CreateAccessKey](#)
 - [CreatePolicy](#)
 - [CreateRole](#)
 - [CreateUser](#)
 - [DeleteAccessKey](#)
 - [DeletePolicy](#)

- [DeleteRole](#)
- [DeleteUser](#)
- [DeleteUserPolicy](#)
- [DetachRolePolicy](#)
- [PutUserPolicy](#)

Kotlin

SDK for 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: String = "{" +
```

```

    "  \"Version\": \"2012-10-17\",\" +
    "  \"Statement\": [\" +
    "    {\" +
    "      \"Effect\": \"Allow\",\" +
    "      \"Action\": [\" +
    "        \"s3:*\"\" +
    "      ],\" +
    "      \"Resource\": \"*\"\" +
    "    }\" +
    "  ]\" +
    "]"

val request = CreatePolicyRequest {
    policyName = policyNameVal
    policyDocument = policyDocumentValue
}

IamClient { 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 { 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 {
        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 {
    credentialsProvider = staticCredentials
    region = "us-east-1"
}

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 { 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 { 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)
}
```

- Lihat detail API di topik-topik berikut dalam Referensi API AWS SDK For Kotlin.
 - [AttachRolePolicy](#)
 - [CreateAccessKey](#)

- [CreatePolicy](#)
- [CreateRole](#)
- [CreateUser](#)
- [DeleteAccessKey](#)
- [DeletePolicy](#)
- [DeleteRole](#)
- [DeleteUser](#)
- [DeleteUserPolicy](#)
- [DetachRolePolicy](#)
- [PutUserPolicy](#)

PHP

SDK for PHP

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya 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";

```

- Lihat detail API di topik-topik berikut dalam Referensi API AWS SDK for PHP .

- [AttachRolePolicy](#)
- [CreateAccessKey](#)
- [CreatePolicy](#)
- [CreateRole](#)
- [CreateUser](#)
- [DeleteAccessKey](#)
- [DeletePolicy](#)
- [DeleteRole](#)
- [DeleteUser](#)
- [DeleteUserPolicy](#)
- [DetachRolePolicy](#)
- [PutUserPolicy](#)

Python

SDK for 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 kredensial 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()
```

- Lihat detail API di topik-topik berikut dalam Referensi API AWS SDK for Python (Boto3).
 - [AttachRolePolicy](#)
 - [CreateAccessKey](#)
 - [CreatePolicy](#)
 - [CreateRole](#)
 - [CreateUser](#)
 - [DeleteAccessKey](#)
 - [DeletePolicy](#)
 - [DeleteRole](#)
 - [DeleteUser](#)
 - [DeleteUserPolicy](#)
 - [DetachRolePolicy](#)
 - [PutUserPolicy](#)

Ruby

SDK for 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 kredensial 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
```

```
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
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)
end

```

```
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__
```

- Lihat detail API di topik-topik berikut dalam Referensi API AWS SDK for 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 pada aplikasi yang berjalan di instans Amazon EC2.

Aplikasi yang berjalan pada instans Amazon EC2 harus menyertakan AWS kredensial dalam permintaan API. AWS Anda dapat meminta pengembang menyimpan AWS kredensial langsung dalam instans Amazon EC2 dan mengizinkan aplikasi dalam hal itu untuk menggunakan kredensial

tersebut. Tetapi pengembang kemudian harus mengelola kredensial dan memastikan bahwa mereka meneruskan kredensial dengan aman ke setiap instance dan memperbarui setiap instans Amazon EC2 ketika saatnya untuk memperbarui kredensialnya. Banyak pekerjaan tambahan.

Sebagai gantinya, Anda dapat dan harus menggunakan peran IAM untuk mengelola kredensial sementara untuk aplikasi yang berjalan pada instans Amazon EC2. Bila Anda menggunakan peran, Anda tidak perlu mendistribusikan kredensial jangka panjang (seperti kredensial masuk atau kunci akses) ke instans Amazon EC2. Sebaliknya, peran menyediakan izin sementara yang dapat digunakan aplikasi saat mereka melakukan panggilan ke sumber daya AWS lainnya. Saat meluncurkan instans Amazon EC2, Anda menentukan peran IAM untuk diasosiasikan 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 diabstraksikan AWS oleh sistem operasi tervirtualisasi. Karena pemisahan ekstra ini, Anda memerlukan langkah tambahan untuk menetapkan AWS peran dan izin terkait ke instans Amazon EC2 dan membuatnya tersedia untuk aplikasinya. Langkah ekstra ini adalah pembuatan [profil instance](#) yang dilampirkan 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 sekaligus, dan semua aplikasi pada instans memiliki 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 dijalankannya. Menetapkan setiap tugas peran selaras dengan prinsip akses yang paling tidak memiliki hak 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 Amazon Elastic Container Service.

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 satu

peran untuk beberapa instance, Anda dapat membuat perubahan pada satu peran tersebut dan perubahan tersebut menyebar secara otomatis ke semua instance.

Note

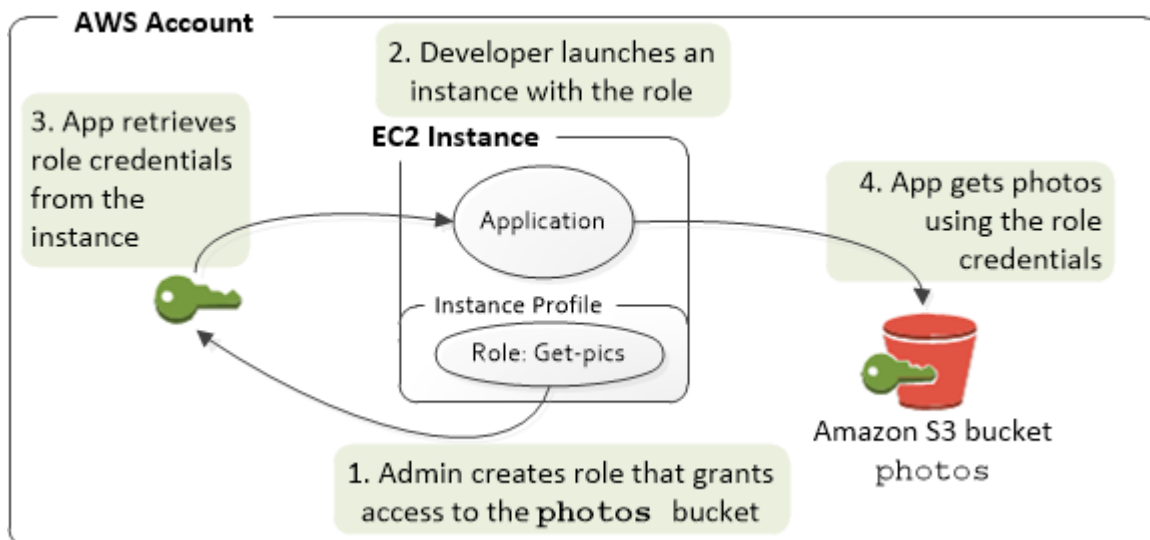
Meskipun peran biasanya ditetapkan ke instans Amazon EC2 saat Anda meluncurkannya, peran juga dapat dilampirkan ke instans Amazon EC2 yang sedang berjalan. Untuk mempelajari cara melampirkan peran pada instans yang sedang berjalan, lihat [Peran IAM untuk Amazon EC2](#).

Topik

- [Bagaimana cara kerja peran untuk instans Amazon EC2?](#)
- [Izin yang diperlukan untuk menggunakan peran dengan Amazon EC2](#)
- [Bagaimana saya memulainya?](#)
- [Informasi terkait](#)
- [Menggunakan profil instans](#)

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. 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 kredensi 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 menetapkan bahwa hanya instans Amazon EC2 yang dapat mengambil peran tersebut. Dalam kebijakan izin peran, administrator menentukan izin hanya-baca untuk bucket photos.
2. Pengembang meluncurkan instans Amazon EC2 dan menetapkan peran Get-pics tersebut 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 atau untuk membuat dan mengelola peran dan instans Amazon EC2, 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 [metadana instans](#) Amazon EC2, sebagaimana dijelaskan dalam [Mengambil Kredensial Keamanan dari Metadana 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 SDK AWS individual menjelaskan fitur yang didukung oleh SDK tersebut guna mengelola kredensial.)

Atau, aplikasi bisa mendapatkan kredensi sementara langsung dari metadata instance 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.

Kredensi keamanan sementara yang tersedia pada instans secara otomatis diperbarui sebelum kedaluwarsa sehingga set yang valid selalu tersedia. Aplikasi hanya perlu memastikan bahwa itu mendapatkan serangkaian kredensial baru dari metadata instans sebelum metadata yang sedang digunakan kedaluwarsa. Anda dapat menggunakan SDK AWS 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 kategori. `iam/security-credentials/role-name`

Izin yang diperlukan untuk menggunakan peran dengan Amazon EC2

Untuk meluncurkan instance dengan peran, pengembang harus memiliki izin untuk meluncurkan instans Amazon EC2 dan izin untuk meneruskan peran IAM.

Kebijakan sampel berikut menzinkan pengguna menggunakan AWS Management Console untuk meluncurkan instans dengan peran. Kebijakan ini mencakup wildcard (*) untuk memungkinkan pengguna meneruskan peran apa pun dan melakukan tindakan Amazon EC2 yang terdaftar. `ListInstanceProfilesTindakan` 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.

```
{
  "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 instance. 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 diteruskannya ke instans Amazon EC2 memiliki izin untuk bekerja dengan IAM dan Amazon DynamoDB. Dalam hal ini, Alice mungkin dapat meluncurkan instance, masuk ke dalamnya, mendapatkan kredensial keamanan sementara, dan kemudian melakukan tindakan IAM atau DynamoDB yang tidak dia otorisasi.

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 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 diperiksa AWS setiap kali ARN peran diteruskan sebagai sebuah parameter ke API (atau konsol melakukannya 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 kepada pengguna untuk meluncurkan instans Amazon EC2 dengan peran tertentu

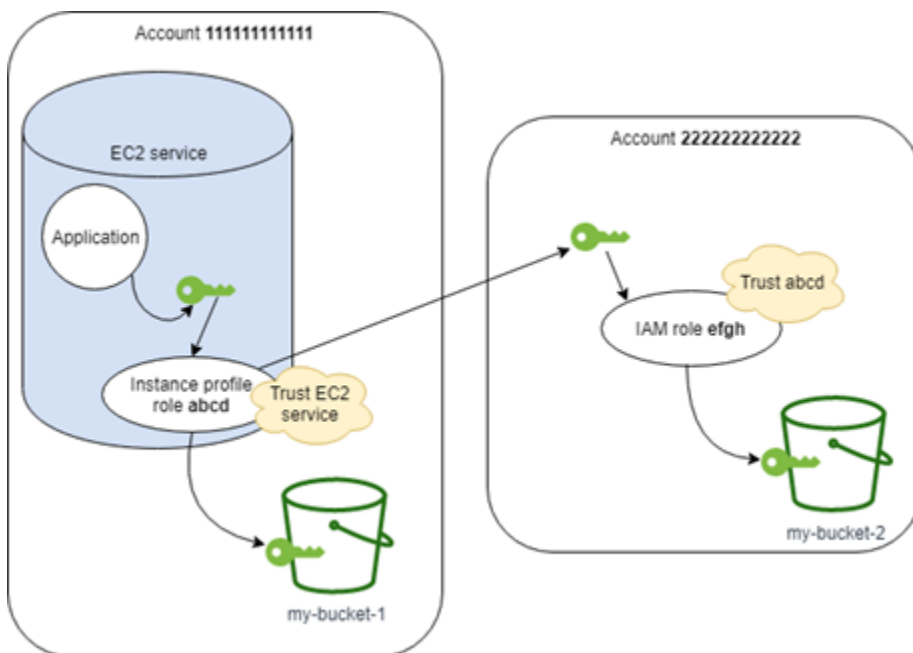
Kebijakan sampel berikut memungkinkan pengguna untuk menggunakan Amazon EC2 API untuk meluncurkan instans dengan peran. Elemen Resource menentukan Amazon Resource Name (ARN) dari peran. Dengan menentukan ARN, kebijakan memberi izin kepada pengguna untuk hanya meneruskan peran Get-pics. Jika pengguna 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.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "iam:PassRole",
      "Resource": "arn:aws:iam::account-id: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 instans Amazon EC2 ada di akun. 111111111111 Instance tersebut menyertakan peran profil `abcd` instance yang memungkinkan aplikasi melakukan tugas Amazon S3 hanya-baca di bucket dalam akun `my-bucket-1` yang sama. 111111111111 Namun demikian, aplikasi tersebut juga harus diizinkan untuk menjalankan `efgh` peran lintas akun untuk mengakses `my-bucket-2` bucket Amazon S3 dalam akun 222222222222.



Peran profil instans `abcd` Amazon EC2 harus memiliki kebijakan izin berikut untuk mengizinkan aplikasi mengakses bucket Amazon S3 `my-bucket-1`:

Akun 111111111111 Kebijakan ***abcd*** Izin Peran

```
{
  "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:::my-bucket-1/*",
        "arn:aws:s3:::my-bucket-1"
    ]
},
{
    "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 **abcd** Kepercayaan Peran

```

{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "abcdTrustPolicy",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"Service": "ec2.amazonaws.com"}
        }
    ]
}

```

```
}

```

Anggap bahwa `efgh` peran lintas akun memungkinkan tugas Amazon S3 hanya baca di bucket `my-bucket-2` dengan akun `222222222222` yang sama. Untuk melakukannya, peran lintas akun `efgh` harus memiliki kebijakan izin berikut:

Akun `222222222222` Kebijakan ***efgh*** Izin Peran

```
{
  "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:::my-bucket-2/*",
        "arn:aws:s3:::my-bucket-2"
      ]
    }
  ]
}
```

Peran `efgh` harus mempercayai profil instans `abcd` untuk menjalankannya. Untuk melakukannya, peran `efgh` harus memiliki kebijakan kepercayaan berikut:

Akun `222222222222` Kebijakan ***efgh*** Kepercayaan Peran

```
{

```

```
"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 penelusuran yang menunjukkan aplikasi yang berjalan pada instans Amazon EC2 yang menggunakan kredensi sementara untuk peran guna membaca bucket Amazon S3. Setiap penelusuran berikut menyajikan langkah serupa dengan bahasa pemrograman yang berbeda:
 - [Konfigurasi Peran IAM untuk Amazon EC2 dengan SDK for Java di Panduan Pengembang AWS SDK for Java](#)
 - [Luncurkan Instans Amazon EC2 menggunakan SDK for .NET di Panduan Pengembang AWS SDK for .NET](#)
 - [Membuat Instans Amazon EC2 dengan SDK for Ruby di Panduan Pengembang AWS SDK for Ruby](#)

Informasi terkait

Untuk informasi selengkapnya tentang membuat peran atau peran untuk instans Amazon EC2, lihat informasi berikut:

- Untuk informasi lebih lanjut, lihat [Menggunakan peran IAM dengan instans Amazon EC2](#), ke Panduan Pengguna Amazon EC2 untuk Instans Linux.

- Untuk membuat peran, lihat [Membuat 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 [Menggunakan profil instans](#).
- Untuk informasi selengkapnya tentang kredensial keamanan sementara untuk peran dalam instans metadata, lihat [Mengambil Kredensial Keamanan dari Metadata Instans](#) dalam Panduan Pengguna Amazon EC2 untuk Instans Linux.

Menggunakan profil instans

Gunakan profil instans untuk meneruskan peran IAM ke instans EC2. Untuk informasi lebih lanjut, lihat [Peran IAM untuk Amazon EC2](#) dalam Panduan Pengguna Amazon EC2 untuk Instans Linux.

Mengelola profil instans (konsol)

Jika Anda menggunakan AWS Management Console untuk membuat peran Amazon EC2, konsol akan secara otomatis membuat profil instans 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 instans. 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 AWS Management Console 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 [Menghapus profil peran atau instans](#).

Mengelola profil instance (AWS CLI atau AWS API)

Jika Anda mengelola peran Anda dari AWS CLI atau AWS API, Anda membuat peran dan profil 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 instans, untuk mengidentifikasi, mengatur, dan mengontrol akses ke mereka. Anda dapat menandai profil instance hanya ketika Anda menggunakan AWS CLI atau AWS API.

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 [konsistensi 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 instans (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 instans (API AWS)

Anda dapat memanggil operasi AWS API berikut untuk bekerja dengan profil instance di file 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](#)

Mencabut kredensial 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 AWS Management Console 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, lihat [Menonaktifkan 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 sebuah peran.

Untuk segera menolak semua izin untuk pengguna kredensial peran saat ini

1. Masuk ke AWS Management Console 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. centang kotak 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 [aws: 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:TokenIssueTimeNilai` hanya ada dalam konteks AWS permintaan yang ditandatangani dengan kredensial keamanan sementara, sehingga pernyataan Deny dalam kebijakan tidak memengaruhi permintaan yang ditandatangani dengan kredensial 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.

```
{
  "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
```

Mengelola peran IAM

Kadang Anda perlu mengubah atau menghapus peran yang telah Anda buat. Untuk mengubah peran, Anda dapat melakukan hal-hal berikut:

- Mengubah kebijakan yang terkait dengan peran tersebut
- Ubah siapa yang dapat mengakses peran
- Ubah izin yang diberikan peran tersebut kepada pengguna
- Mengubah pengaturan durasi sesi maksimum untuk peran yang diambil menggunakan AWS Management Console, AWS CLI atau API

Anda juga dapat menghapus peran yang tidak diperlukan lagi. Anda dapat mengelola peran dari AWS Management Console, AWS CLI, dan API.

Topik

- [Mengubah peran](#)
- [Menghapus profil peran atau instans](#)

Mengubah peran

Anda dapat menggunakan AWS Management Console, AWS CLI, atau IAM API untuk membuat perubahan pada peran.

Topik

- [Lihat akses peran](#)
- [Menghasilkan kebijakan berdasarkan informasi akses](#)
- [Mengubah peran \(konsol\)](#)
- [Mengubah peran \(AWS CLI\)](#)
- [Mengubah peran \(API AWS\)](#)

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 [Menyaring izin di AWS menggunakan informasi yang terakhir diakses](#).

Menghasilkan kebijakan berdasarkan informasi akses

Terkadang Anda dapat memberikan izin ke entitas IAM (pengguna atau peran) melampaui izin yang mereka butuhkan. Untuk membantu menyempurnakan izin yang diberikan, 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 Anda tentukan. Anda dapat menggunakan templat untuk membuat kebijakan terkelola dengan izin terperinci dan kemudian melampirkannya ke IAM entitas. Dengan begitu, Anda hanya memberikan izin yang pengguna atau peran perlu berinteraksi dengan sumber daya AWS untuk kasus penggunaan spesifik Anda. Untuk mempelajari informasi lebih lanjut, lihat [Hasilkan kebijakan berdasarkan aktivitas akses](#).

Mengubah peran (konsol)

Anda dapat menggunakan AWS Management Console untuk mengubah peran. Untuk mengubah rangkaian tanda pada peran, lihat [Mengelola tag pada peran IAM \(konsol\)](#).

Topik

- [Mengubah kebijakan kepercayaan peran \(konsole\)](#)
- [Mengubah kebijakan izin peran \(konsol\)](#)
- [Mengubah deskripsi peran \(konsol\)](#)
- [Mengubah durasi sesi maksimum peran \(konsol\)](#)
- [Mebgubah batas izin peran \(konsol\)](#)

Mengubah kebijakan kepercayaan peran (konsole)

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 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 kredensial ini. Ini memastikan bahwa Anda selalu memiliki seperangkat kredensial 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 mengubah kebijakan kepercayaan peran (konsole)

1. Masuk ke AWS Management Console 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 Trust relationship, lalu pilih Edit trust policy.
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 elemen: `Principal`

```
"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 [Memberikan izin pengguna untuk berganti peran](#).

Snippet kebijakan berikut ini menunjukkan cara merujuk dua layanan AWS dalam elemen `Principal`:

```
"Principal": {  
  "Service": [  
    "opsworks.amazonaws.com",  
    "ec2.amazonaws.com"  
  ]  
},
```

6. Setelah selesai mengedit kebijakan kepercayaan, pilih **Perbarui kebijakan** untuk menyimpan perubahan.

Untuk informasi selengkapnya tentang struktur dan sintaks, lihat [Kebijakan dan Izin di IAM](#) 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 [Memberikan izin pengguna untuk berganti peran](#).

1. Masuk ke eksternal terpercaya 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 yang dikelola AWS. Kebijakan yang dikelola AWS muncul dengan ikon AWS



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 perubahan kebijakan.


Mengubah kebijakan izin peran (konsol)

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 (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 tab Izin.
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 yang dikelola AWS.

Kebijakan yang dikelola AWS muncul dengan ikon AWS



Untuk informasi selengkapnya tentang perbedaan antara kebijakan yang dikelola AWS dan kebijakan yang dikelola oleh pelanggan, lihat [Kebijakan 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.

Mengubah deskripsi peran (konsol)

Untuk mengubah deskripsi peran, ubah teks deskripsi.

Untuk mengubah deskripsi peran (konsol)

1. Masuk ke AWS Management Console 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.

Mengubah durasi sesi maksimum peran (konsol)

Untuk menentukan pengaturan durasi sesi maksimum untuk peran yang diasumsikan dengan menggunakan konsol, AWS CLI, atau API AWS, ubah nilai pengaturan 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 layanan AWS.

Untuk mengubah pengaturan durasi sesi maksimum untuk peran yang diasumsikan menggunakan konsol, AWS CLI, atau API AWS (konsol)

1. Masuk ke AWS Management Console 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 kredensial keamanan sementara peran IAM](#).

Dalam AWS Management Console, 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 mengasumsikan peran dari AWS CLI atau API AWS dapat meminta sesi yang lebih lama, hingga maksimal. Pengaturan `MaxSessionDuration` menentukan durasi maksimum dari sesi peran yang dapat diminta.

- Untuk menentukan durasi sesi menggunakan AWS CLI, gunakan parameter `duration-seconds`. Untuk pelajari selengkapnya, lihat [Beralih ke peran IAM \(AWS CLI\)](#).
- Untuk menentukan durasi sesi menggunakan API AWS, gunakan parameter `DurationSeconds`. Untuk pelajari selengkapnya, lihat [Beralih ke peran IAM \(AWS API\)](#).

Membgubah batas izin peran (konsol)

Untuk mengubah izin maksimum yang diperbolehkan untuk peran, ubah [batas izin](#) peran.

Untuk mengubah kebijakan yang digunakan untuk mengatur batas izin untuk peran

1. Masuk ke AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Di panel navigasi, silakan 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.

Mengubah peran (AWS CLI)

Anda dapat menggunakan AWS Command Line Interface untuk mengubah peran. Untuk mengubah rangkaian tanda pada peran, lihat [Mengelola tag pada peran IAM \(AWS CLI atau AWS API\)](#).

Topik

- [Mengubah kebijakan kepercayaan peran \(AWS CLI\)](#)
- [Mengubah kebijakan izin peran \(AWS CLI\)](#)
- [Mengubah deskripsi peran \(AWS CLI\)](#)
- [Mengubah durasi sesi maksimum peran \(AWS CLI\)](#)
- [Mengubah batas izin peran \(AWS CLI\)](#)

Mengubah kebijakan kepercayaan peran (AWS CLI)

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 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 kredensial ini. Ini memastikan bahwa Anda selalu memiliki seperangkat kredensial 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. [Mengubah kebijakan kepercayaan peran \(konsole\)](#)

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:
 - [aws iam daftar peran](#)
2. (Opsional) Untuk melihat penjelasan peran terkini, gunakan perintah berikut:
 - [aws iam 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 Akun AWS mereferensikan dua Principal elemen. Hal ini memungkinkan pengguna dalam dua terpisah Akun AWS untuk mengambil peran ini.

```
{
  "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 [Memberikan izin pengguna untuk berganti peran](#).

4. Untuk menggunakan file yang baru saja Anda buat untuk memperbarui kebijakan kepercayaan, jalankan perintah berikut:
 - [aws iam update-assume-role-policy](#)

Untuk mengizinkan pengguna dalam akun eksternal terpercaya untuk menggunakan peran (AWS CLI)

Untuk informasi dan detail selengkapnya tentang prosedur ini, lihat [Memberikan izin pengguna untuk berganti peran](#).

1. Buat file JSON yang berisi kebijakan izin yang memberikan izin untuk mengasumsikan peran tersebut. Misalnya, kebijakan berikut memuat izin minimum yang diperlukan:

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::ACCOUNT-ID-THAT-CONTAINS-ROLE: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 membuat 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:
 - [aws iam daftar-pengguna](#)
 - [grup daftar aws iam](#)
4. Gunakan salah satu perintah berikut untuk melampirkan kebijakan yang Anda buat pada langkah sebelumnya kepada pengguna atau kelompok:
 - [aws iam attach-user-policy](#)
 - [aws iam attach-group-policy](#)

Mengubah 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](#)

Mengubah deskripsi peran (AWS CLI)

Untuk mengubah deskripsi peran, ubah teks deskripsi.

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](#)

Mengubah durasi sesi maksimum peran (AWS CLI)

Untuk menentukan pengaturan durasi sesi maksimum untuk peran yang diasumsikan menggunakan AWS CLI atau API, ubah nilai pengaturan 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 layanan AWS.

Note

Siapa pun yang mengasumsikan peran dari AWS CLI atau API dapat menggunakan parameter CLI `duration-seconds` atau parameter API `DurationSeconds` untuk meminta sesi 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 pengaturan 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 kredensial keamanan sementara peran IAM](#).

Mengubah batas izin peran (AWS CLI)

Untuk mengubah izin maksimum yang diperbolehkan untuk peran, ubah [batas izin](#) peran.

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.

Mengubah peran (API AWS)

Anda dapat menggunakan API AWS untuk mengubah peran. Untuk mengubah rangkaian tanda pada peran, lihat [Mengelola tag pada peran IAM \(AWS CLI atau AWS API\)](#).

Topik

- [Mengubah kebijakan kepercayaan peran \(API AWS\)](#)
- [Mengubah kebijakan izin peran \(API AWS\)](#)
- [Mengubah deskripsi peran \(API AWS\)](#)
- [Mengubah durasi sesi maksimum peran \(API AWS\)](#)
- [Mengubah batas izin peran \(API AWS\)](#)

Mengubah kebijakan kepercayaan peran (API AWS)

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 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 kredensial sementara dan secara otomatis memperbarui kredensial ini. Ini memastikan bahwa Anda selalu memiliki seperangkat kredensial yang valid. Untuk layanan ini, tidak perlu mengambil peran saat ini lagi untuk mendapatkan kredensial 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 [Mengubah kebijakan kepercayaan peran \(konsole\)](#)

Untuk mengubah kebijakan kepercayaan peran (API AWS)

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 Akun AWS mereferensikan dua `Principal` elemen. Hal ini memungkinkan pengguna dalam dua terpisah Akun AWS untuk mengambil peran ini.

```
{
  "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 [Memberikan izin pengguna untuk berganti peran](#).

4. Untuk menggunakan file yang baru saja Anda buat untuk memperbarui kebijakan kepercayaan, panggil operasi berikut:
 - [UpdateAssumeRolePolicy](#)

Untuk mengizinkan pengguna dalam akun eksternal tepercaya untuk menggunakan peran (API AWS)

Untuk informasi dan detail selengkapnya tentang prosedur ini, lihat [Memberikan izin pengguna untuk berganti peran](#).

1. Buat file JSON yang berisi kebijakan izin yang memberikan izin untuk mengasumsikan peran tersebut. Misalnya, kebijakan berikut memuat izin minimum yang diperlukan:

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::ACCOUNT-ID-THAT-CONTAINS-ROLE: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](#)

Mengubah kebijakan izin peran (API AWS)

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 (API AWS)

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](#)

Mengubah deskripsi peran (API AWS)

Untuk mengubah deskripsi peran, ubah teks deskripsi.

Untuk mengubah deskripsi peran (API AWS)

1. (Opsional) Untuk melihat deskripsi terkini untuk peran, panggil operasi berikut:
 - [GetRole](#)
2. Untuk memperbarui deskripsi peran, panggil operasi berikut dengan parameter deskripsi:

- [UpdateRole](#)

Mengubah durasi sesi maksimum peran (API AWS)

Untuk menentukan pengaturan durasi sesi maksimum untuk peran yang diasumsikan menggunakan AWS CLI atau API, ubah nilai pengaturan 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 layanan AWS.

Note

Siapa pun yang mengasumsikan peran dari AWS CLI atau API dapat menggunakan parameter CLI `duration-seconds` atau parameter API `DurationSeconds` untuk meminta sesi 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 pengaturan durasi sesi maksimum untuk peran yang diasumsikan menggunakan API (API AWS)

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-sessionduration` 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 kredensial keamanan sementara peran IAM](#).

Mengubah batas izin peran (API AWS)

Untuk mengubah izin maksimum yang diperbolehkan untuk peran, ubah [batas izin](#) peran.

Untuk mengubah kebijakan terkelola yang digunakan untuk mengatur batas izin 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.

Menghapus profil peran atau instans

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 aksesnya ditolak ke semua 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 kredensial keamanan sementara peran IAM](#).

Topik

- [Lihat akses peran](#)
- [Menghapus peran yang terkait dengan layanan](#)
- [Menghapus peran IAM \(konsol\)](#)
- [Menghapus peran IAM \(AWS CLI\)](#)
- [Menghapus peran IAM \(AWSAPI\)](#)
- [Informasi terkait](#)

Lihat akses peran

Sebelum menghapus peran, kami menyarankan agar Anda meninjau kapan peran tersebut terakhir kali digunakan. Anda dapat melakukan ini dengan menggunakan AWS Management Console, yang AWS CLI, atau AWSAPI. Anda harus melihat informasi ini karena Anda tidak ingin menghapus akses dari seseorang menggunakan peran tersebut.

Tanggal aktivitas terakhir peran mungkin tidak sesuai dengan tanggal terakhir yang dilaporkan dalam tab Penasihat Akses. Tab [Penasihat Akses](#) melaporkan aktivitas hanya untuk layanan yang diizinkan oleh kebijakan izin peran tersebut. Tanggal aktivitas terakhir peran mencakup upaya terakhir untuk mengakses layanan apa pun di AWS.

Note

Periode pelacakan untuk aktivitas terakhir peran dan data Access Advisor adalah untuk 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 [Lokasi AWS melacak informasi yang terakhir diakses](#).

Untuk melihat kapan peran terakhir digunakan (konsol)

1. Masuk ke AWS Management Console 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 `objekRoleLastUsed`. 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 menampilkan waktu terakhir peran digunakan (API AWS)

[GetRole](#) - Panggil operasi ini untuk mengembalikan informasi tentang peran, termasuk `objekRoleLastUsed`. 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

Jika peran tersebut adalah [peran yang terkait dengan layanan](#), tinjau dokumentasi untuk layanan yang terkait 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. Untuk informasi selengkapnya, lihat [Menghapus peran yang ditautkan ke layanan](#).

Menghapus peran IAM (konsol)

Bila Anda menggunakan AWS Management Console untuk menghapus peran, IAM secara otomatis melepaskan kebijakan terkelola yang terkait dengan peran tersebut. Ini juga secara otomatis menghapus kebijakan inline apa pun yang terkait dengan peran tersebut, 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. Jika demikian, Anda dapat menggunakan AWS Management Console untuk menghapus peran dan profil instans. 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. Sebaliknya, Anda harus menggunakan AWS CLI, Alat untuk Windows PowerShell, atau AWS API untuk pertama menghapus peran dari profil instans. Kemudian, Anda harus melakukan langkah terpisah untuk menghapus peran tersebut.

Untuk menghapus peran (konsol)

1. Masuk ke AWS Management Console 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. Dalam kotak dialog konfirmasi, tinjau informasi yang terakhir diakses, yang menunjukkan kapan terakhir kali setiap peran yang dipilih mengakses layanan AWS. 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.

ℹ 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 instans tanpa menghapus peran, Anda harus menggunakan AWS CLI atau API AWS. Untuk informasi lebih lanjut, lihat bagian berikut.

Menghapus peran IAM (AWS CLI)

Bila 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 instans 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 instans 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 yang dikelola 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 (AWSAPI)

Saat Anda menggunakan IAM API 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 instans terkait yang berisi peran, Anda harus menghapusnya secara terpisah.

Untuk menghapus peran (API AWS)

1. Untuk mencantumkan semua profil instans yang terkait dengan peran, panggil [ListInstanceProfilesForRole](#).

Untuk menghapus peran dari profil instans, 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, panggil [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 dilampirkan pada peran, hubungi [DetachRolePolicy](#). Anda harus meneruskan nama peran dan kebijakan terkelola ARN.

4. Panggil [DeleteRole](#) untuk menghapus peran.

Informasi terkait

Untuk informasi umum tentang profil instans, lihat [Menggunakan profil instans](#).

Untuk informasi umum tentang peran yang terkait dengan layanan, lihat [Menggunakan peran terkait layanan](#).

Penyedia dan federasi identitas

Jika Anda sudah mengelola identitas pengguna di luar AWS, 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 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 web yang memerlukan akses ke sumber daya AWS .

IdP eksternal menyediakan informasi identitas untuk AWS menggunakan OpenID [Connect \(OIDC\)](#) atau [SAMP 2.0 \(Security Assertion Markup Language 2.0\)](#). OIDC menghubungkan aplikasi, seperti GitHub Actions, yang tidak berjalan AWS ke AWS sumber daya. Contoh penyedia identitas SAMP yang terkenal adalah Shibboleth dan Layanan Federasi Direktori Aktif.

Note

Sebagai praktik terbaik keamanan, kami sarankan Anda mengelola pengguna manusia di [IAM Identity Center dengan penyedia identitas](#) SAMP eksternal alih-alih menggunakan federasi SAMP di IAM. Untuk informasi tentang situasi tertentu di mana pengguna IAM diperlukan, lihat [Kapan membuat pengguna IAM \(bukan peran\)](#).

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 Akun AWS keamanan Anda karena Anda tidak perlu mendistribusikan atau menanamkan kredensial keamanan jangka panjang, seperti kunci akses, dalam aplikasi Anda.

Panduan ini mencakup federasi IAM. Kasus penggunaan Anda mungkin lebih didukung oleh IAM Identity Center atau Amazon Cognito. Ringkasan dan tabel berikut memberikan gambaran umum tentang metode yang dapat digunakan pengguna Anda untuk mendapatkan akses gabungan ke sumber daya. AWS

	Jenis akun	Manajemen akses..	Sumber identitas yang didukung
Federasi dengan Pusat Identitas IAM	Beberapa akun yang dikelola oleh AWS Organizations	Pengguna manusia tenaga kerja Anda	<ul style="list-style-type: none"> • SAML 2.0 • Direktori Aktif Terkelola • Direktori Pusat Identitas
Federasi dengan IAM	Akun tunggal dan mandiri	<ul style="list-style-type: none"> • Pengguna manusia dalam penerapan jangka pendek dan skala kecil • Pengguna mesin 	<ul style="list-style-type: none"> • SAML 2.0 • OIDC
Federasi dengan kumpulan identitas Amazon Cognito	Setiap	Pengguna aplikasi yang memerlukan otorisasi IAM untuk mengakses sumber daya	<ul style="list-style-type: none"> • SAML 2.0 • OIDC • Pilih penyedia identitas sosial OAuth 2.0

Federasi dengan Pusat Identitas IAM

Untuk manajemen akses terpusat pengguna manusia, kami menyarankan Anda menggunakan [IAM Identity Center](#) untuk mengelola akses ke akun dan izin Anda dalam akun tersebut. Pengguna di Pusat Identitas IAM diberikan kredensial jangka pendek ke sumber daya Anda. AWS Anda dapat menggunakan Active Directory, penyedia identitas eksternal (iDP), atau direktori IAM Identity Center sebagai sumber identitas bagi pengguna dan grup untuk menetapkan akses ke sumber daya Anda.

AWS

IAM Identity Center mendukung federasi identitas dengan SAMP (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 SAMP, termasuk aplikasi pihak ketiga, seperti Microsoft 365, SAP Concur, AWS Management Console dan Salesforce.

Federasi dengan IAM

Meskipun kami sangat menyarankan untuk mengelola pengguna manusia di IAM Identity Center, Anda dapat mengaktifkan akses pengguna federasi dengan IAM untuk pengguna manusia dalam penerapan skala kecil jangka pendek. IAM memungkinkan Anda untuk menggunakan SAMP 2.0 dan Open ID Connect (OIDC) yang terpisah IdPs dan menggunakan atribut pengguna federasi untuk kontrol akses. Dengan IAM, Anda dapat meneruskan atribut pengguna, seperti pusat biaya, judul, atau lokal, dari IdPs ke Anda AWS, dan menerapkan izin akses berbutir halus berdasarkan atribut ini.

Beban kerja adalah kumpulan sumber daya dan kode yang memberikan nilai bisnis, seperti aplikasi atau proses backend. Beban kerja Anda dapat 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 AWS Lambda atau 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 kredensial 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 di seluruh Akun AWS, lihat [Tutorial IAM: Delegasikan akses AWS Akun yang menggunakan peran IAM](#).

Untuk menautkan IDP 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 [SAMP 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 mengautentikasi 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 kredensial IAM yang memberi pengguna akses ke sumber daya terlindungi yang Anda kelola. AWS Identity pool memperoleh kredensi untuk sesi sementara melalui operasi [AssumeRoleWithWebIdentity](#) API.

Amazon Cognito bekerja dengan penyedia identitas eksternal yang mendukung SAMP 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.

Skenario umum

Note

Kami menyarankan Anda meminta pengguna manusia Anda untuk menggunakan kredensi sementara saat mengakses. AWS Sudahkah Anda mempertimbangkan untuk menggunakan AWS IAM Identity Center? Anda dapat menggunakan Pusat Identitas IAM untuk mengelola akses ke beberapa secara terpusat Akun AWS dan memberi pengguna akses masuk tunggal yang dilindungi MFA ke semua akun yang ditetapkan dari satu tempat. Dengan IAM Identity Center, Anda dapat membuat dan mengelola identitas pengguna di IAM Identity Center atau dengan mudah terhubung ke penyedia identitas kompatibel SAMP 2.0 yang ada. Untuk informasi lebih lanjut, lihat [Apa itu Pusat Identitas IAM?](#) dalam AWS IAM Identity Center User Guide.

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 (SAMP). 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 menginformasikan AWS tentang iDP eksternal dan konfigurasinya. Ini membangun kepercayaan antara IDP Anda Akun AWS dan eksternal. Topik berikut menyediakan skenario umum untuk menggunakan penyedia identitas IAM.

Topik

- [Menggunakan Amazon Cognito untuk aplikasi seluler](#)
- [Menggunakan operasi API federasi OIDC untuk aplikasi seluler](#)

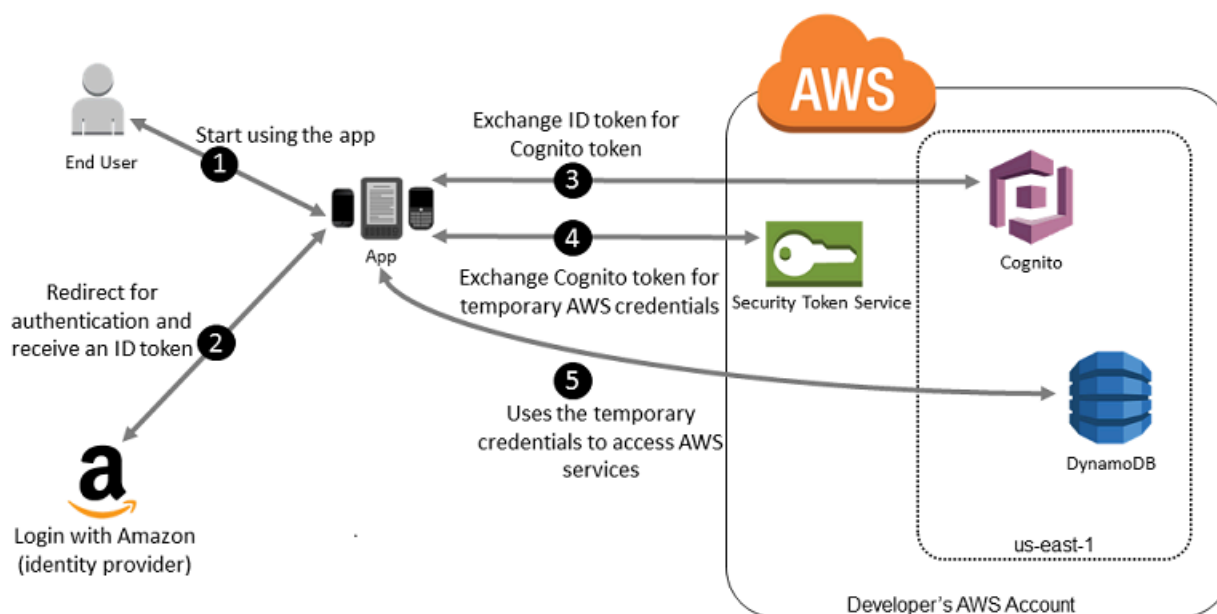
Menggunakan Amazon Cognito untuk aplikasi seluler

Cara yang lebih disukai untuk menggunakan federasi OIDC adalah dengan menggunakan Amazon [Cognito](#). Misalnya, Adele 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 untuk 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. Dalam dirinya Akun AWS yang berisi bucket Amazon S3 dan tabel DynamoDB untuk game, Adele menggunakan Amazon Cognito untuk membuat peran IAM yang secara tepat menentukan izin yang dibutuhkan game. [Jika dia menggunakan OIDC iDP, dia juga menciptakan entitas penyedia identitas IAM OIDC untuk membangun kepercayaan antara kumpulan identitas Amazon Cognito dalam 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 serangkaian kredensi keamanan sementara yang terdiri dari ID kunci AWS akses, kunci akses rahasia, dan token sesi. Aplikasi kemudian dapat menggunakan kredensi 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 IdP apa pun yang kompatibel dengan OIDC, 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 API Amazon Cognito `GetId` dan `GetCredentialsForIdentity` untuk menukar token Login with Amazon ID dengan token Amazon Cognito. Amazon Cognito, yang telah dikonfigurasi untuk mempercayai proyek Login with Amazon Anda, menghasilkan token yang ditukar dengan kredensi sesi sementara. AWS STS
4. Aplikasi ini menerima kredensi keamanan sementara dari Amazon Cognito. Aplikasi Anda juga dapat menggunakan alur kerja Dasar (Klasik) di Amazon Cognito untuk mengambil token dari penggunaan. AWS STS `AssumeRoleWithWebIdentity` Untuk informasi selengkapnya, lihat

[Alur autentikasi kumpulan identitas \(identitas gabungan\)](#) 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 kredensial keamanan sementara dan kebijakan yang ditetapkan menentukan apa yang dapat diakses.

Gunakan proses berikut untuk mengonfigurasi aplikasi agar menggunakan Amazon Cognito guna mengautentikasi pengguna dan memberikan akses aplikasi ke resource. AWS Untuk langkah-langkah khusus untuk menyelesaikan skenario ini, lihat dokumentasi untuk Amazon Cognito.

1. (Opsional) Daftar sebagai pengembang dengan Login with Amazon, Facebook, Google, atau OpenID Connect (OIDC) lainnya yang kompatibel dengan IDP dan konfigurasi satu atau beberapa aplikasi dengan penyedia. Langkah ini bersifat opsional karena Amazon Cognito juga mendukung akses tidak terotentikasi (tamu) untuk pengguna Anda.
2. Pergi ke [Amazon Cognito di](#). AWS Management Console 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) peran yang Anda kaitkan dengan kumpulan identitas. Wizard Amazon Cognito di AWS Management Console menyediakan kode contoh untuk membantu Anda memulai.
5. Saat aplikasi Anda mengakses AWS resource, teruskan instance penyedia kredensial ke objek klien, yang meneruskan kredensial keamanan sementara ke klien. Izin untuk kredensial didasarkan pada peran atau peran-peran yang Anda tetapkan sebelumnya.

Untuk informasi selengkapnya, lihat hal berikut:

- [Masuk \(Android\)](#) di Dokumentasi AWS Amplify Kerangka.
- [Masuk \(iOS\)](#) di Dokumentasi AWS Amplify Kerangka.

Menggunakan operasi API 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 identifikasi 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 AWS Management Console. 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 untuknya.
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 diambil oleh aplikasi jika pengguna masuk melalui Login with

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 (seperti Amazon.com) sebagai Principal (entitas tepercaya), dan sertakan Condition yang sesuai dengan ID aplikasi yang ditetapkan IdP. Contoh peran untuk penyedia yang berbeda dijelaskan dalam [Membuat peran untuk Penyedia Identitas pihak ketiga \(federasi\)](#).

4. Dalam aplikasi, autentikasi pengguna Anda dengan IdP. Spesifikasi tentang cara melakukan ini bervariasi baik menurut IDP yang Anda gunakan (Login with Amazon, Facebook, atau Google) dan di platform mana aplikasi Anda berjalan. Misalnya, metode otentikasi aplikasi Android dapat berbeda dari metode aplikasi iOS atau aplikasi web JavaScript berbasis.

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 autentikasi IDP dan menentukan Amazon Resource Name (ARN) untuk peran IAM yang Anda buat untuk IDP tersebut. AWS memverifikasi bahwa token tersebut tepercaya dan valid dan jika demikian, mengembalikan kredensi 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 kredensial 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 dalam folder Amazon S3 yang menyertakan ID pengguna sebagai awalan atau sufiks. Hal ini memungkinkan Anda membuat kebijakan kontrol akses yang mengunci folder sehingga hanya pengguna dengan ID tersebut yang tidak dapat mengaksesnya. Untuk informasi lebih lanjut, lihat [Identifikasi pengguna dengan federasi OIDC](#) dalam topik ini.
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

`keAssumeRoleWithWebIdentity` , karena token IdP juga biasanya kedaluwarsa setelah waktu tetap. [Jika Anda menggunakan AWS SDK for iOS atau AWS SDK for Android, Anda dapat menggunakan tindakan `CredentialsProvider Amazonsts`, yang mengelola kredensial sementara IAM, termasuk menyegarkannya sesuai kebutuhan.](#)

Federasi OIDC

Bayangkan Anda membuat aplikasi yang mengakses AWS sumber daya, seperti GitHub Tindakan yang menggunakan 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 dalam aplikasi di luar. AWS sebagai gantinya, konfigurasi aplikasi Anda untuk meminta kredensial AWS keamanan sementara secara dinamis saat diperlukan menggunakan federasi OIDC. Kredensial sementara yang disediakan memetakan ke AWS 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 GitHub Actions atau IdP yang kompatibel dengan OpenID [Connect \(OIDC\) lainnya, untuk mengautentikasi](#). AWS Mereka menerima token otentikasi, yang dikenal sebagai JSON Web Token (JWT), dan kemudian menukar token itu untuk 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 Akun AWS keamanan karena Anda tidak perlu menanamkan dan mendistribusikan kredensial keamanan jangka panjang dengan aplikasi Anda.

Dalam sebagian besar skenario, kami menyarankan agar Anda menggunakan [Amazon Cognito](#) karena sebagai pialang identitas dan melakukan banyak pekerjaan federasi untuk Anda. Untuk detailnya, lihat bagian berikut, [Menggunakan Amazon Cognito untuk aplikasi seluler](#).

Note

JSON Web Tokens (JWTs) yang dikeluarkan oleh penyedia identitas OpenID Connect (OIDC) berisi waktu kedaluwarsa dalam klaim yang menentukan kapan token kedaluwarsa `exp`. IAM menyediakan jendela lima menit di luar waktu kedaluwarsa yang ditentukan dalam JWT untuk memperhitungkan kemiringan jam, seperti yang diizinkan oleh standar OpenID Connect ([OIDC](#)) Core 1.0. Ini berarti JWT OIDC diterima oleh IAM

setelah waktu kedaluwarsa tetapi dalam jendela lima menit ini diterima untuk evaluasi dan pemrosesan lebih lanjut.

Topik

- [Buat penyedia identitas OpenID Connect \(OIDC\) di IAM](#)
- [Dapatkan cap jempol untuk penyedia identitas OpenID Connect](#)
- [Identifikasi pengguna dengan federasi OIDC](#)
- [Sumber daya tambahan untuk federasi OIDC](#)

Buat penyedia identitas OpenID Connect (OIDC) di IAM

Penyedia identitas IAM OIDC adalah entitas dalam 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 IDP yang kompatibel dengan OIDC 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 AWS Management Console, Tools for Windows AWS Command Line Interface PowerShell, atau IAM API.

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 ditetapkan secara dinamis ke pengguna gabungan 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 dilakukan oleh pengguna federasi. AWS Untuk membuat peran bagi penyedia identitas pihak ketiga, lihat [Membuat peran untuk Penyedia Identitas pihak ketiga \(federasi\)](#).

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 itu di `oidc-provider` ARN sebagai nilai elemen. Resource Anda

juga memiliki opsi untuk menambahkan garis miring dan wildcard (/*) ke domain URL atau menggunakan karakter wildcard (*dan?) di titik mana pun di jalur URL. Jika URL penyedia identitas OIDC dalam permintaan tidak cocok dengan nilai yang ditetapkan dalam Resource elemen kebijakan, permintaan gagal.

Topik

- [Membuat dan mengelola penyedia OIDC \(konsol\)](#)
- [Membuat dan mengelola penyedia identitas IAM OIDC \(AWS CLI\)](#)
- [Membuat dan mengelola Penyedia Identitas OIDC \(API\)AWS](#)

Membuat dan mengelola penyedia OIDC (konsol)

Ikuti petunjuk ini untuk membuat dan mengelola penyedia identitas IAM OIDC di AWS Management Console

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 [Buat 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 beberapa penyedia identitas OIDC (IdPs) melalui perpustakaan otoritas sertifikat root tepercaya (CA) kami alih-alih menggunakan cap jempol sertifikat untuk memverifikasi sertifikat server IDP Anda. Dalam kasus ini,

sidik jari lama Anda tetap ada dalam konfigurasi Anda, tetapi tidak lagi digunakan untuk validasi. OIDC ini IdPs termasuk Auth0, GitHub, Google GitLab, dan yang menggunakan bucket Amazon S3 untuk meng-host titik akhir JSON Web Key Set (JWKS).

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.
6. Pilih Dapatkan sidik jari untuk memverifikasi sertifikat server IdP. Untuk mempelajari caranya, lihat [Dapatkan cap jempol untuk penyedia identitas OpenID Connect](#).

Note

Rantai 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 duplikat atau sertifikat tambahan, maka Anda menerima kesalahan ketidakcocokan tanda tangan dan STS gagal memvalidasi JSON Web Token (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](#).

7. Untuk Audiens, ketikkan ID klien dari aplikasi yang Anda daftarkan dengan IDP dan terima [Step 1](#), dan yang membuat permintaan. AWS Jika memiliki ID klien tambahan (juga dikenal sebagai audiens) untuk IdP ini, Anda dapat menambahkannya nanti di halaman detail penyedia.
8. (Opsional) Untuk Menambahkan tag, Anda dapat menambahkan pasangan kunci-nilai untuk membantu Anda mengidentifikasi dan mengatur. IdPs Anda juga dapat menggunakan tanda untuk mengontrol akses ke sumber daya AWS. Untuk mempelajari lebih lanjut tentang menandai penyedia identitas IAM OIDC, lihat [Menandai penyedia identitas OpenID Connect \(OIDC\)](#) Pilih Tambahkan tanda. Masukkan nilai untuk setiap pasangan nilai-kunci tanda.
9. Verifikasi informasi yang telah Anda berikan. Setelah selesai, pilih Tambahkan penyedia.


10. Tetapkan peran IAM ke penyedia identitas Anda untuk memberikan identitas pengguna eksternal yang dikelola oleh izin penyedia identitas Anda 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 \(federasi\)](#).

 Note

OIDC yang IdPs digunakan dalam kebijakan kepercayaan peran harus dalam akun yang sama dengan peran yang mempercayainya.

Untuk menambahkan atau menghapus sidik jari untuk penyedia identitas IAM OIDC (konsol)

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. Di bagian Sidik jari pilih Kelola. 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 dari aplikasi yang Anda daftarkan dengan IDP dan diterima [Step 1](#), dan yang akan membuat permintaan. AWS 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 konfirmasi 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. Di panel navigasi, pilih Penyedia identitas.
3. Pilih kotak centang di samping 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 (API)AWS

Anda dapat menggunakan perintah IAM API berikut untuk membuat dan mengelola penyedia OIDC.

Untuk membuat penyedia identitas IAM OIDC (API)AWS

1. (Opsional) Untuk mendapatkan daftar semua penyedia identitas IAM OIDC di AWS akun Anda, hubungi 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 (API) IAM OIDC yang ada AWS

- Untuk memperbarui daftar sidik jari sertifikat server untuk penyedia identitas IAM OIDC, panggil operasi berikut:
 - [UpdateOpenIDConnectProviderThumbprint](#)

Untuk menandai penyedia identitas IAM OIDC (API) yang ada AWS

- Untuk menandai penyedia identitas IAM OIDC yang ada, hubungi operasi berikut:
 - [TagOpenIDConnectProvider](#)

Untuk mencantumkan tag untuk penyedia identitas (API) IAM OIDC yang ada AWS

- Untuk mencantumkan tag untuk penyedia identitas IAM OIDC yang ada, panggil operasi berikut:
 - [ListOpenIDConnectProviderTags](#)

Untuk menghapus tag pada penyedia identitas (API) IAM OIDC yang ada AWS

- Untuk menghapus tag pada penyedia identitas IAM OIDC yang ada, hubungi operasi berikut:
 - [UntagOpenIDConnectProvider](#)

Untuk menambah atau menghapus ID klien dari penyedia identitas (API) IAM OIDC yang ada AWS

1. (Opsional) Untuk mendapatkan daftar semua penyedia identitas IAM OIDC di AWS akun Anda, hubungi 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 (API) IAM OIDC AWS

1. (Opsional) Untuk mendapatkan daftar semua penyedia identitas IAM OIDC di AWS akun Anda, hubungi 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 cap jempol untuk penyedia identitas OpenID Connect

Saat Anda [membuat penyedia identitas OpenID Connect \(OIDC\)](#) di konsol IAM, Anda harus menyediakan cap jempol. IAM memerlukan cap jempol untuk otoritas sertifikat menengah atas (CA) yang menandatangani sertifikat yang digunakan oleh penyedia identitas eksternal (IdP). Sidik jari adalah tanda tangan untuk sertifikat CA yang digunakan untuk menerbitkan sertifikat bagi IdP yang kompatibel dengan OIDC. Saat Anda membuat penyedia identitas IAM OIDC, Anda mempercayai identitas yang diautentikasi oleh IDP tersebut untuk memiliki akses ke identitas Anda. Akun AWS Dengan memasok sidik jari sertifikat CA, Anda percaya pada sertifikat apa pun yang dikeluarkan oleh CA 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 akar 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, [Tools for Windows AWS Command Line Interface PowerShell](#), atau IAM API. Bila Anda menggunakan metode ini, sidik jari adalah opsional. Jika Anda memilih untuk tidak menyertakan sidik jari, IAM akan mengambil cap jempol CA menengah atas dari sertifikat server OIDC 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, konsol](#) mencoba mengambil cap jempol untuk Anda. Kami menyarankan Anda juga mendapatkan sidik jari untuk OIDC iDP Anda secara manual dan memverifikasi bahwa konsol mengambil sidik jari yang benar. Untuk informasi selengkapnya tentang mendapatkan cap jempol sertifikat, lihat bagian berikut.

Dapatkan cap jempol sertifikat

Anda menggunakan browser web dan alat baris perintah OpenSSL untuk mendapatkan cap jempol sertifikat untuk penyedia OIDC. Namun, Anda tidak perlu mendapatkan cap jempol sertifikat secara manual untuk membuat penyedia identitas IAM OIDC. Anda dapat menggunakan prosedur berikut untuk mendapatkan cap jempol sertifikat dari penyedia OIDC Anda.

Note

AWS mengamankan komunikasi dengan beberapa penyedia identitas OIDC (IdPs) melalui perpustakaan otoritas sertifikat root tepercaya (CA) kami alih-alih menggunakan cap jempol sertifikat untuk memverifikasi sertifikat server IDP Anda. Dalam kasus ini, sidik jari lama Anda tetap ada dalam konfigurasi Anda, tetapi tidak lagi digunakan untuk validasi. OIDC ini IdPs termasuk Auth0,,, Google GitHub GitLab, dan mereka yang menggunakan bucket Amazon S3 untuk meng-host titik akhir JSON Web Key Set (JWKS).

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",
  ...
}
```

4. Gunakan alat baris perintah OpenSSL untuk menjalankan perintah berikut. Ganti `keys.example.com` dengan nama domain yang Anda peroleh di [Step 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 menengah teratas dalam rantai otoritas sertifikat.

```

-----BEGIN CERTIFICATE-----
MIICiTCCAFICCCQD6m7oRw0uX0jANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6
b24xFDASBgNVBA5TC01BTSBDb25zb2x1MRIwEAYDVQQDEw1UZXR0Q21sYWxhZAd
BgkqhkiG9w0BCQEWEG5vb251QGftYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD
VQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBA5TC01BTSBDb25z
b2x1MRIwEAYDVQQDEw1UZXR0Q21sYWxhZAdBgkqhkiG9w0BCQEWEG5vb251QGft
YXpvbi5jb20wZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
rDHudUZg3qX4waLG5M43q7Wgc/MbQITx0USQv7c7ugFFDzQGBzZswY6786m86gpE
Ibb30hjZnzcvcQAaRHhd1QWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
nUHVvXyUntneD9+h8Mg9q6q+auNKyExzyLwax1Aoo7TJHidbtS4J5iNmZgXL0Fkb
FFBjvSfpJI1J00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp3780D8uTs7fLvjx79LjSTb
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

Rantai 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 duplikat atau sertifikat tambahan, maka Anda menerima kesalahan ketidakcocokan tanda tangan dan STS gagal memvalidasi JSON Web Token (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](#).

- 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
```

7. Jika Anda membuat penyedia identitas IAM OIDC dengan AWS CLI, Tools for Windows, atau IAM API PowerShell, berikan cap jempol ini saat membuat penyedia.

Jika Anda membuat penyedia identitas IAM OIDC di konsol IAM, bandingkan sidik jari ini dengan sidik jari yang ditunjukkan di halaman konsol Verifikasi Informasi Penyedia saat Anda membuat penyedia OIDC.

Important

Jika sidik jari yang Anda dapatkan tidak cocok dengan yang Anda lihat di konsol, Anda tidak boleh membuat penyedia OIDC di konsol. Sebaliknya, Anda harus menunggu kemudian mencoba kembali untuk membuat penyedia OIDC, memastikan bahwa sidik jari cocok sebelum Anda membuat 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. Pergi ke [OpenSSL: Sumber](https://openssl.org/source/), Tarballs (<https://openssl.org/source/>).
2. Unduh sumber terbaru dan buat paketnya.

Untuk menginstal OpenSSL di Windows

1. Buka [OpenSSL: Distribusi Biner](https://wiki.openssl.org/index.php/Binaries) (<https://wiki.openssl.org/index.php/Binaries>) 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. Installer 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. Mengatur 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 di 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 Pengaturan sistem lanjutan, tab Lanjutan, Variabel Lingkungan.) Untuk informasi selengkapnya, lihat dokumentasi Windows.

Identifikasi pengguna dengan federasi OIDC

Saat Anda membuat kebijakan akses di IAM, seringkali berguna untuk dapat menentukan izin berdasarkan aplikasi yang dikonfigurasi dan pada ID pengguna yang telah mengautentikasi menggunakan penyedia identitas eksternal (IdP). Misalnya, aplikasi seluler Anda yang menggunakan federasi OIDC dapat menyimpan informasi di Amazon S3 menggunakan struktur seperti ini:

```
myBucket/app1/user1  
myBucket/app1/user2  
myBucket/app1/user3  
...  
myBucket/app2/user1  
myBucket/app2/user2
```

```
myBucket/app2/user3
```

```
...
```

Anda mungkin juga ingin membedakan secara tambahan jalur ini dari penyedia. Dalam hal ini, struktur mungkin terlihat seperti berikut (hanya dua penyedia yang terdaftar untuk menyimpan ruang):

```
myBucket/Amazon/app1/user1
```

```
myBucket/Amazon/app1/user2
```

```
myBucket/Amazon/app1/user3
```

```
...
```

```
myBucket/Amazon/app2/user1
```

```
myBucket/Amazon/app2/user2
```

```
myBucket/Amazon/app2/user3
```

```
myBucket/Facebook/app1/user1
```

```
myBucket/Facebook/app1/user2
```

```
myBucket/Facebook/app1/user3
```

```
...
```

```
myBucket/Facebook/app2/user1
```

```
myBucket/Facebook/app2/user2
```

```
myBucket/Facebook/app2/user3
```

```
...
```

Untuk struktur ini, app1 dan app2 mewakili aplikasi yang berbeda, seperti game yang berbeda, dan setiap pengguna aplikasi memiliki folder yang berbeda. Nilai untuk app1 dan app2 bisa berupa nama bersahabat yang Anda tetapkan (misalnya, mynumbergame) atau mungkin merupakan ID aplikasi yang ditetapkan penyedia saat Anda mengonfigurasi aplikasi. Jika Anda memutuskan untuk memasukkan nama penyedia di jalur tersebut, itu juga dapat berupa nama ramah seperti Cognito, Amazon, Facebook, dan Google.

Anda biasanya dapat membuat folder untuk app1 dan app2 melalui AWS Management Console, karena nama aplikasi adalah nilai statis. Hal ini juga berlaku jika Anda menyertakan nama penyedia dalam jalur, karena nama penyedia juga merupakan nilai statis. Sebaliknya, folder khusus pengguna (*user1*, *user2*, *user3*, dll.) harus dibuat pada saat waktu aktif dari aplikasi, menggunakan ID pengguna yang tersedia di nilai SubjectFromWebIdentityToken yang dikembalikan oleh permintaan ke AssumeRoleWithWebIdentity.

Untuk menulis kebijakan yang memungkinkan akses eksklusif ke sumber daya bagi pengguna individu, Anda dapat mencocokkan nama folder lengkap, termasuk nama aplikasi dan nama

penyedia, jika Anda menggunakannya. Kemudian, Anda dapat menyertakan kunci konteks spesifik penyedia layanan yang merujuk ID pengguna yang dikembalikan oleh penyedia:

- `cognito-identity.amazonaws.com:sub`
- `www.amazon.com:user_id`
- `graph.facebook.com:id`
- `accounts.google.com:sub`

Untuk penyedia OIDC, gunakan URL yang sepenuhnya memenuhi syarat dari penyedia OIDC dengan kunci subkonteks, seperti contoh berikut:

- `server.example.com:sub`

Contoh berikut menunjukkan kebijakan izin yang memberikan akses ke bucket di Amazon S3 hanya jika awalan untuk bucket sesuai dengan rangkaian:

`myBucket/Amazon/mynumbersgame/user1`

Contoh ini mengasumsikan bahwa pengguna masuk menggunakan Login with Amazon, dan pengguna menggunakan aplikasi yang dipanggil `mynumbersgame`. ID unik pengguna ditampilkan sebagai atribut yang disebut `user_id`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["s3:ListBucket"],
      "Resource": ["arn:aws:s3:::myBucket"],
      "Condition": {"StringLike": {"s3:prefix": ["Amazon/mynumbersgame/
${www.amazon.com:user_id}/*"]}}
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject",
        "s3:DeleteObject"
      ],
      "Resource": [
```

```
        "arn:aws:s3:::myBucket/amazon/mynumbersgame/${www.amazon.com:user_id}",
        "arn:aws:s3:::myBucket/amazon/mynumbersgame/${www.amazon.com:user_id}/*"
    ]
}
]
```

Anda akan membuat kebijakan serupa untuk pengguna yang masuk menggunakan Amazon Cognito, Facebook, Google, atau IDP yang kompatibel dengan OpenID Connect lainnya. Kebijakan tersebut akan menggunakan nama penyedia yang berbeda sebagai bagian dari jalur serta ID aplikasi yang berbeda.

Untuk informasi selengkapnya tentang kunci federasi OIDC yang tersedia untuk pemeriksaan kondisi dalam kebijakan, lihat. [Kunci yang tersedia untuk AWS federasi OIDC](#)

Sumber daya tambahan untuk federasi OIDC

Sumber daya berikut dapat membantu Anda mempelajari lebih lanjut tentang federasi OIDC:

- Menggunakan OpenID Connect dalam GitHub alur kerja Anda dengan Mengonfigurasi [OpenID Connect](#) di Amazon Web Services
- [Identitas Amazon Cognito](#) di Amplify Libraries for Android Guide dan Amazon [Cognito Identity di Amplify Libraries for Swift Guide](#).
- [Mengotomatisasi Peran Identitas Web IAM AWS Berbasis Koneksi OpenID dengan Microsoft Entra ID](#) di AWS blog Jaringan Mitra (APN) berjalan melalui cara mengautentikasi proses latar belakang otomatis atau aplikasi yang berjalan di luar menggunakan otorisasi OIDC. AWS machine-to-machine
- Artikel [Federasi Identitas Web dengan Aplikasi Seluler](#) membahas federasi OIDC dan menunjukkan contoh bagaimana menggunakan federasi OIDC untuk mendapatkan akses ke konten di Amazon S3.

Federasi SAML 2.0

AWS mendukung federasi identitas dengan [SAMP 2.0 \(Security Assertion Markup Language 2.0\)](#), standar terbuka yang digunakan oleh banyak penyedia identitas (). IdPs Fitur ini memungkinkan sistem masuk tunggal gabungan (SSO), sehingga pengguna dapat masuk ke AWS Management Console atau memanggil operasi AWS API tanpa Anda harus membuat pengguna IAM untuk semua orang di organisasi Anda. Dengan menggunakan SAMP, Anda dapat menyederhanakan proses

konfigurasi federasi dengan AWS, karena Anda dapat menggunakan layanan IDP alih-alih [menulis kode proxy identitas kustom](#).

Federasi IAM mendukung penggunaan kasus ini:

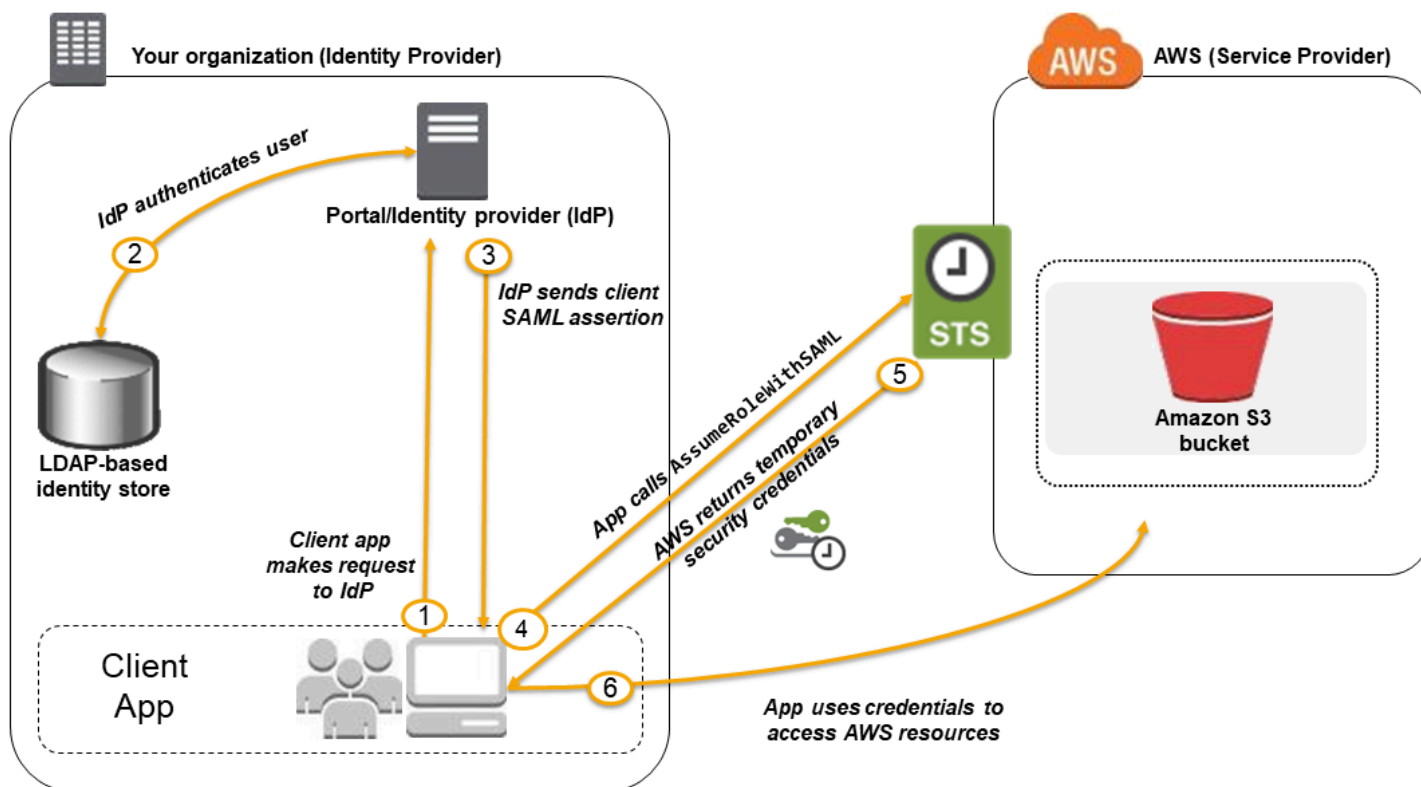
- [Akses gabungan untuk memungkinkan pengguna atau aplikasi di organisasi Anda memanggil operasi AWS API](#). Anda menggunakan pernyataan SAML (sebagai bagian dari tanggapan autentikasi) yang dibuat dalam organisasi Anda untuk mendapatkan kredensial keamanan sementara. Skenario ini serupa dengan skenario federasi lain yang mendukung IAM, seperti yang dijelaskan dalam [Meminta kredensial keamanan sementara](#) dan [Federasi OIDC](#). Namun, SAMP 2.0 yang berbasis IdPs di organisasi Anda menangani banyak detail saat dijalankan untuk melakukan otentikasi dan pemeriksaan otorisasi. Ini adalah skenario yang dibahas dalam topik ini.
- [Single sign-on \(SSO\) berbasis web ke dari AWS Management Console](#) organisasi Anda. Pengguna dapat masuk ke portal di organisasi Anda yang dihosting oleh IDP yang kompatibel dengan SAMP 2.0, memilih opsi untuk pergi ke, dan diarahkan AWS ke konsol tanpa harus memberikan informasi login 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 [Mengaktifkan akses broker identitas khusus ke konsol AWS](#).

Topik

- [Menggunakan federasi berbasis SAML untuk akses API ke AWS](#)
- [Ikhtisar pengkonfigurasian federasi berbasis SAML 2.0](#)
- [Ikhtisar peran untuk memungkinkan akses federasi SAML ke sumber daya Anda AWS](#)
- [Mengidentifikasi pengguna secara unik dalam federasi berbasis SAML](#)
- [Buat penyedia identitas SAMP di IAM](#)
- [Konfigurasi IDP SAMP 2.0 Anda dengan mengandalkan kepercayaan pihak dan menambahkan klaim](#)
- [Integrasikan penyedia solusi SAMP pihak ketiga dengan AWS](#)
- [Konfigurasi pernyataan SAMP untuk respons otentikasi](#)
- [Mengaktifkan pengguna gabungan SAML 2.0 untuk mengakses AWS Management Console](#)

Menggunakan federasi berbasis SAML 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 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.
4. Aplikasi klien memanggil AWS STS [AssumeRoleWithSAML](#) API, meneruskan ARN dari penyedia SAMP, ARN peran yang akan diambil, dan pernyataan SAMP dari iDP.
5. Tanggapan API ke aplikasi klien meliputi kredensial keamanan sementara.
6. Aplikasi klien menggunakan kredensial keamanan sementara untuk menghubungi operasi API Amazon S3.

Ikhtisar pengkonfigurasi federasi berbasis SAML 2.0

Sebelum Anda dapat menggunakan federasi berbasis SAMP 2.0 seperti yang dijelaskan dalam skenario dan diagram sebelumnya, Anda harus mengonfigurasi IDP organisasi Anda dan Anda untuk saling mempercayai. Akun AWS Proses umum untuk mengonfigurasi 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, kami menyarankan Anda mengonfigurasi IDP dan AWS federasi Anda untuk mendukung beberapa titik akhir masuk SAMP. Untuk detailnya, lihat artikel Blog AWS Keamanan [Cara menggunakan endpoint SAMP regional untuk failover](#).

Untuk mengonfigurasi IDP organisasi Anda dan AWS saling mempercayai

1. Daftar AWS sebagai penyedia layanan (SP) dengan IDP organisasi Anda. Gunakan dokumen metadata SAMP dari [https://*region-code*.signin.aws.amazon.com/static/saml-metadata.xml](https://<i>region-code</i>.signin.aws.amazon.com/static/saml-metadata.xml)

Untuk daftar kemungkinan nilai *kode wilayah*, lihat kolom Wilayah di titik akhir [AWS Masuk](#).

Anda dapat secara opsional menggunakan dokumen metadata SAMP dari <https://signin.aws.amazon.com/static/saml-metadata.xml>

2. Menggunakan iDP organisasi Anda, Anda menghasilkan file XML metadata setara yang dapat menggambarkan IDP Anda sebagai penyedia identitas IAM di AWS. Ini harus menyertakan nama penerbit, tanggal pembuatan, tanggal kedaluwarsa, dan kunci yang AWS dapat digunakan untuk memvalidasi tanggapan otentikasi (pernyataan) dari organisasi Anda.
3. Di konsol IAM, Anda membuat entitas penyedia identitas SAML. Sebagai bagian dari proses ini, Anda mengunggah dokumen metadata SAML yang dibuat oleh IdP di organisasi Anda di [Step 2](#). Untuk informasi selengkapnya, lihat [Buat penyedia identitas SAMP di IAM](#).
4. Di IAM, Anda membuat satu atau lebih peran IAM. Dalam kebijakan kepercayaan peran, Anda menetapkan penyedia SAMP sebagai prinsipal, yang membangun 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 \(federasi\)](#).

Note

IDP SAMP yang digunakan dalam kebijakan kepercayaan peran harus berada dalam akun yang sama dengan peran tersebut.

5. Di iDP organisasi Anda, Anda menentukan pernyataan yang memetakan pengguna atau grup di 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 [Konfigurasi pernyataan SAMP untuk respons otentikasi](#).

Jika iDP Anda mengaktifkan SSO ke AWS konsol, maka Anda dapat mengonfigurasi durasi maksimum sesi konsol. Untuk informasi selengkapnya, lihat [Mengaktifkan pengguna gabungan SAML 2.0 untuk mengakses AWS Management Console](#).

Note

AWS Implementasi federasi SAMP 2.0 tidak mendukung pernyataan SAMP terenkripsi antara penyedia identitas IAM dan. AWS Namun, lalu lintas antara sistem pelanggan dan AWS ditransmisikan melalui saluran terenkripsi (TLS).

6. Dalam aplikasi yang Anda buat, Anda memanggil AWS Security Token Service AssumeRoleWithSAML API, meneruskannya ARN dari penyedia SAMP yang Anda buat, ARN peran untuk berasumsi bahwa Anda membuat [Step 4](#), dan pernyataan SAMP tentang pengguna saat ini yang Anda dapatkan dari idP Anda. [Step 3](#) AWS memastikan bahwa permintaan untuk mengambil peran berasal dari iDP yang direferensikan di penyedia SAMP.

Untuk informasi selengkapnya, lihat [AssumeRoleWithSAMP](#) di Referensi AWS Security Token Service API.

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 akses federasi SAML ke sumber daya Anda AWS

Peran atau peran yang Anda buat di IAM menentukan apa yang diizinkan dilakukan oleh pengguna federasi 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 menyebutkan bahwa hanya pengguna yang afiliasi SAML-nya `staff` (seperti yang dinyatakan oleh `https://openidp.feide.no`) diizinkan untuk mengakses peran tersebut, sebagaimana digambarkan oleh kebijakan sampel berikut:

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Principal": {"Federated": "arn:aws:iam::account-id:saml-provider/ExampleOrgSSOProvider"},
    "Action": "sts:AssumeRoleWithSAML",
    "Condition": {
      "StringEquals": {
        "saml:aud": "https://signin.aws.amazon.com/saml",
        "saml:iss": "https://openidp.feide.no"
      },
      "ForAllValues:StringLike": {"saml:edupersonaffiliation": ["staff"]}
    }
  }]
}
```

Note

IDP SAMP yang digunakan dalam kebijakan kepercayaan peran harus berada dalam akun yang sama dengan peran tersebut.

Untuk informasi selengkapnya tentang kunci SAML yang dapat Anda periksa di kebijakan, lihat [Kunci yang tersedia untuk federasi AWS STS berbasis SAML](#).

Anda dapat menyertakan titik akhir regional untuk `saml:aud` atribut di `https://region-code.signin.aws.amazon.com/static/saml-metadata.xml`. Untuk daftar kemungkinan nilai *kode wilayah*, lihat kolom Wilayah di titik akhir [AWS Masuk](#).

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 mengelola instans Amazon Elastic Compute Cloud, Anda harus secara eksplisit mengizinkan tindakan Amazon EC2 dalam kebijakan izin, seperti yang ada dalam kebijakan terkelola AmazonEC2. FullAccess

Mengidentifikasi pengguna secara unik dalam federasi berbasis SAML

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:

```
myBucket/app1/user1
myBucket/app1/user2
myBucket/app1/user3
```

Anda dapat membuat bucket (myBucket) dan folder (app1) melalui konsol Amazon S3 atau AWS CLI, karena itu adalah nilai statis. Namun, folder khusus pengguna (*user1*, *user2*, *user3*, dll.) harus dibuat pada waktu pengoperasian menggunakan kode, karena nilai yang mengidentifikasi pengguna tidak diketahui hingga 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 SAMP 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 federasi AWS STS berbasis SAML](#).

Kombinasi dari NameQualifier dan Subject dapat digunakan untuk secara unik mengidentifikasi pengguna federasi. Kode pseudo berikut menunjukkan bagaimana nilai ini dihitung. Dalam kode pseudo ini + menunjukkan kontegasi, SHA1 mewakili fungsi yang menghasilkan rangkuman pesan menggunakan SHA-1, dan Base64 mewakili fungsi yang menghasilkan versi yang dienkodkan Base-64 dari output hash.

```
Base64 ( SHA1 ( "https://example.com/saml" + "123456789012" + "/"
MySAMLIdP" ) )
```

Untuk informasi lebih lanjut tentang kunci kebijakan yang tersedia untuk federasi SAML, lihat [Kunci yang tersedia untuk federasi AWS STS berbasis SAML](#).

- `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 [Konfigurasi pernyataan SAMP untuk respons otentikasi](#).

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 dan. `saml:namequalifier` `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.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "s3:GetObject",
      "s3:PutObject",
      "s3:DeleteObject"
    ],
    "Resource": [
      "arn:aws:s3:::exampleorgBucket/backup/${saml:namequalifier}/${saml:sub}",
      "arn:aws:s3:::exampleorgBucket/backup/${saml:namequalifier}/${saml:sub}/*"
    ],
    "Condition": {"StringEquals": {"saml:sub_type": "persistent"}}
  }
}
```

```
}
```

Untuk informasi selengkapnya tentang pernyataan pemetaan dari IdP ke kunci kebijakan, lihat [Konfigurasi pernyataan SAMP untuk respons otentikasi](#).

Buat penyedia identitas SAMP 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 saat ingin membangun kepercayaan antara IDP yang kompatibel dengan SAMP seperti Shibboleth atau Layanan Federasi Direktori Aktif AWS dan, sehingga pengguna di organisasi 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 AWS Management Console atau dengan AWS CLI, Alat untuk 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). Namun dalam konteks ini, peran ditetapkan secara dinamis ke pengguna gabungan 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 dilakukan oleh pengguna federasi. AWS Untuk menciptakan peran bagi federasi SAML, lihat [Membuat peran untuk Penyedia Identitas pihak ketiga \(federasi\)](#).

Akhirnya, setelah Anda membuat peran, Anda menyelesaikan kepercayaan SAMP dengan mengonfigurasi IDP Anda dengan informasi tentang AWS dan peran yang Anda ingin pengguna federasi Anda gunakan. Ini disebut sebagai mengonfigurasi kepercayaan pihak pengandal antara IdP dan AWS. Untuk mengkonfigurasi kepercayaan pihak yang bergantung, lihat [Konfigurasi IDP SAMP 2.0 Anda dengan mengandalkan kepercayaan pihak dan menambahkan klaim](#).

Topik

- [Membuat dan mengelola penyedia identitas SAMP IAM \(konsol\)](#)
- [Membuat dan mengelola Penyedia Identitas SAMP IAM \(AWS CLI\)](#)
- [Membuat dan mengelola penyedia identitas SAMP IAM \(API\)AWS](#)

Membuat dan mengelola penyedia identitas SAMP IAM (konsol)

Anda dapat menggunakan AWS Management Console untuk membuat dan menghapus penyedia identitas IAM SAMP.

Untuk membuat penyedia identitas IAM SAML (konsol)

1. Sebelum Anda dapat membuat penyedia identitas SAMP IAM, Anda memerlukan dokumen metadata SAMP yang Anda dapatkan dari IdP. Dokumen ini mencakup nama penerbit, informasi kedaluwarsa, dan kunci yang dapat digunakan untuk memvalidasi respons autentikasi SAML (penegasan) yang diterima dari IdP. Untuk membuat dokumen metadata, gunakan perangkat lunak manajemen identitas yang digunakan organisasi Anda sebagai IdP. Untuk petunjuk tentang cara mengonfigurasi banyak yang tersedia IdPs untuk dikerjakan AWS, termasuk cara membuat dokumen metadata SAMP yang diperlukan, lihat [Integrasikan penyedia solusi SAMP pihak ketiga dengan AWS](#)

Important

File metadata ini mencakup nama penerbit, informasi kedaluwarsa, dan kunci yang dapat digunakan untuk memvalidasi respons otentikasi SAMP (pernyataan) yang diterima dari IdP. File metadata harus diekode dalam format UTF-8 tanpa tanda urutan byte (BOM). Untuk menghapus BOM, Anda dapat mengkode file sebagai UTF-8 menggunakan alat bantu edit teks, seperti Notepad++.


Sertifikat x.509 yang disertakan sebagai bagian dari dokumen metadata SAMP harus menggunakan ukuran kunci minimal 1024 bit. Selain itu, sertifikat x.509 juga harus bebas dari ekstensi berulang. Anda dapat menggunakan ekstensi, tetapi ekstensi hanya dapat muncul sekali dalam sertifikat. Jika sertifikat x.509 tidak memenuhi salah satu kondisi, pembuatan IDP gagal dan mengembalikan kesalahan “Tidak dapat mengurai metadata”. Seperti yang didefinisikan oleh [SAMP V2.0 Metadata Interoperability Profile Versi 1.0](#), IAM tidak mengevaluasi atau mengambil tindakan terkait berakhirnya sertifikat X.509 dokumen metadata.

2. Masuk ke AWS Management Console dan 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 SAML.
5. Ketikkan nama untuk penyedia identitas.

6. Untuk Dokumen metadata, pilih **Pilih file**, tentukan dokumen metadata SAML yang Anda unduh di [Step 1](#).
7. (Opsional) Untuk Tambahkan tag, Anda dapat menambahkan pasangan kunci-nilai untuk membantu Anda mengidentifikasi dan mengatur. IdPs Anda juga dapat menggunakan tanda untuk mengontrol akses ke sumber daya AWS . Untuk mempelajari selengkapnya tentang penandaan penyedia identitas, lihat [Menandai penyedia identitas IAM SAM](#).

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 untuk memberikan identitas pengguna eksternal yang dikelola oleh izin penyedia identitas Anda 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 \(federasi\)](#).

 Note

IDP SAMP yang digunakan dalam kebijakan kepercayaan peran harus berada dalam akun yang sama dengan peran tersebut.

Untuk menghapus penyedia SAML (konsol)

1. Masuk ke AWS Management Console 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.

Membuat dan mengelola Penyedia Identitas SAMP IAM (AWS CLI)

Anda dapat menggunakan AWS CLI untuk membuat dan mengelola penyedia SAMP.

Sebelum Anda dapat membuat penyedia identitas IAM, Anda memerlukan dokumen metadata SAMP yang Anda dapatkan dari iDP. Dokumen ini mencakup nama penerbit, informasi kedaluwarsa, dan kunci yang dapat digunakan untuk memvalidasi respons autentikasi SAML (penegasan) yang

diterima dari IdP. Untuk membuat dokumen metadata, gunakan perangkat lunak manajemen identitas yang digunakan organisasi Anda sebagai IdP. Untuk petunjuk tentang cara mengonfigurasi banyak yang tersedia IdPs untuk dikerjakan AWS, termasuk cara membuat dokumen metadata SAMP yang diperlukan, lihat. [Integrasikan penyedia solusi SAMP pihak ketiga dengan AWS](#)

Important

File metadata ini mencakup nama penerbit, informasi kedaluwarsa, dan kunci yang dapat digunakan untuk memvalidasi respons otentikasi SAMP (pernyataan) yang diterima dari IdP. File metadata harus dienkodkan dalam format UTF-8 tanpa tanda urutan byte (BOM). Untuk menghapus BOM, Anda dapat mengkode file sebagai UTF-8 menggunakan alat bantu edit teks, seperti Notepad++.

Sertifikat x.509 yang disertakan sebagai bagian dari dokumen metadata SAMP harus menggunakan ukuran kunci minimal 1024 bit. Selain itu, sertifikat x.509 juga harus bebas dari ekstensi berulang. Anda dapat menggunakan ekstensi, tetapi ekstensi hanya dapat muncul sekali dalam sertifikat. Jika sertifikat x.509 tidak memenuhi salah satu kondisi, pembuatan IDP gagal dan mengembalikan kesalahan “Tidak dapat mengurai metadata”.

Seperti yang didefinisikan oleh [SAMP V2.0 Metadata Interoperability Profile Versi 1.0](#), IAM tidak mengevaluasi atau mengambil tindakan terkait berakhirnya sertifikat X.509 dokumen metadata.

Untuk membuat penyedia identitas IAM dan mengunggah dokumen metadata ()AWS CLI

- Jalankan perintah ini: [aws iam create-saml-provider](#)

Untuk mengunggah dokumen metadata baru untuk penyedia identitas IAM ()AWS CLI

- 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 SAMP IAM (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 spesifik, seperti ARN, tanggal pembuatan, dan kedaluwarsa, 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 SAMP IAM (API)AWS

Anda dapat menggunakan AWS API untuk membuat dan mengelola penyedia SAMP.

Sebelum Anda dapat membuat penyedia identitas IAM, Anda memerlukan dokumen metadata SAMP yang Anda dapatkan dari IdP. Dokumen ini mencakup nama penerbit, informasi kedaluwarsa, dan kunci yang dapat digunakan untuk memvalidasi respons autentikasi SAML (penegasan) yang diterima dari IdP. Untuk membuat dokumen metadata, gunakan perangkat lunak manajemen identitas yang digunakan organisasi Anda sebagai IdP. Untuk petunjuk tentang cara mengonfigurasi banyak yang tersedia IdPs untuk dikerjakan AWS, termasuk cara membuat dokumen metadata SAMP yang diperlukan, lihat. [Integrasikan penyedia solusi SAMP pihak ketiga dengan AWS](#)

Important

File metadata harus diekode dalam format UTF-8 tanpa tanda urutan byte (BOM). Selain itu, sertifikat X.509 yang disertakan sebagai bagian dari dokumen metadata SAML harus menggunakan ukuran kunci minimal 1024 bit. Jika ukuran kunci lebih kecil, pembuatan IdP gagal dengan kesalahan "Tidak dapat menguraikan metadata". Untuk menghapus BOM, Anda dapat mengkode file sebagai UTF-8 menggunakan alat bantu edit teks, seperti Notepad ++.

Untuk membuat penyedia identitas IAM dan mengunggah dokumen metadata (API)AWS

- Hubungi operasi ini: [CreateSAMLProvider](#)

Untuk mengunggah dokumen metadata baru untuk penyedia identitas IAM (API)AWS

- Hubungi operasi ini: [UpdateSAMLProvider](#)

Untuk menandai penyedia identitas IAM (AWS API) yang ada

- Hubungi operasi ini: [TagSAMLProvider](#)

Untuk mencantumkan tag untuk penyedia identitas IAM (AWS API) yang ada

- Hubungi operasi ini: [ListSAMLProviderTags](#)

Untuk menghapus tag pada penyedia identitas IAM (AWS API) yang ada

- 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, hubungi operasi berikut:
 - [ListSAMLProviders](#)
2. (Opsional) Untuk mendapatkan informasi tentang penyedia tertentu, seperti ARN, tanggal pembuatan, dan kedaluwarsa, hubungi operasi berikut:
 - [GetSAMLProvider](#)
3. Untuk menghapus IdP, panggil operasi berikut:
 - [DeleteSAMLProvider](#)

Konfigurasi IDP SAMP 2.0 Anda dengan mengandalkan kepercayaan pihak dan menambahkan klaim

Saat Anda membuat penyedia identitas IAM dan peran untuk akses SAMP, Anda memberi tahu AWS tentang penyedia identitas eksternal (iDP) dan apa yang boleh 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 `https://region-code.signin.aws.amazon.com/static/saml-metadata.xml` atau `https://signin.aws.amazon.com/static/saml-metadata.xml`. Untuk daftar kemungkinan nilai *kode wilayah*, lihat kolom *Region* di titik akhir [AWS Masuk](#).

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 SAMP ke AWS titik akhir, itu termasuk pernyataan SAMP 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 SAMP dari IDP Anda berisi atribut yang diperlukan yang AWS digunakan dalam kebijakan IAM untuk memeriksa izin bagi pengguna federasi. Untuk informasi selengkapnya, lihat topik berikut:

- [Ikhtisar peran untuk memungkinkan akses federasi SAML ke sumber daya Anda AWS](#). Topik ini membahas penggunaan kunci khusus SAML dalam kebijakan IAM dan cara menggunakannya untuk membatasi izin bagi pengguna federasi SAML.
- [Konfigurasi pernyataan SAMP untuk respons otentikasi](#). 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 termasuk dalam pernyataan SAMP dalam bentuk yang AWS dapat mengenali dan menggunakan.
- [Integrasi penyedia solusi SAMP pihak ketiga dengan AWS](#). Topik ini menyediakan tautan ke dokumentasi yang disediakan oleh organisasi pihak ketiga tentang cara mengintegrasikan solusi identitas AWS.

Note

Untuk meningkatkan ketahanan federasi, kami menyarankan Anda mengonfigurasi IDP dan AWS federasi Anda untuk mendukung beberapa titik akhir masuk SAMP. Untuk detailnya, lihat artikel Blog AWS Keamanan [Cara menggunakan endpoint SAMP regional untuk failover](#).

Integrasikan penyedia solusi SAMP pihak ketiga dengan AWS

Note

Kami menyarankan Anda meminta pengguna manusia Anda untuk menggunakan kredensi sementara saat mengakses. AWS Sudahkah Anda mempertimbangkan untuk menggunakan AWS IAM Identity Center? Anda dapat menggunakan Pusat Identitas IAM untuk mengelola akses ke beberapa secara terpusat Akun AWS dan memberi pengguna akses masuk tunggal yang dilindungi MFA ke semua akun yang ditetapkan dari satu tempat. Dengan IAM Identity Center, Anda dapat membuat dan mengelola identitas pengguna di IAM Identity Center atau dengan mudah terhubung ke penyedia identitas yang kompatibel SAMP 2.0 yang ada. Untuk informasi lebih lanjut, lihat [Apa itu Pusat Identitas IAM?](#) dalam AWS IAM Identity Center User Guide.

Tautan berikut membantu Anda mengonfigurasi solusi penyedia identitas SAMP 2.0 (IDP) pihak ketiga untuk bekerja AWS dengan federasi.

Tip

AWS Support engineer 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 di situs web dokumentasi Auth0 ini memiliki tautan ke sumber

Solusi	Informasi lebih lanjut
	<p>daya yang menjelaskan cara mengatur sistem masuk tunggal (SSO) dengan AWS Management Console dan menyertakan contoh. JavaScript Anda dapat mengonfigurasi Auth0 untuk meneruskan tag sesi. Untuk informasi selengkapnya, lihat Auth0 Mengumumkan Kemitraan dengan Tag AWS Sesi IAM.</p>
Microsoft Entra	<p>Tutorial: Integrasi Microsoft Entra SSO dengan AWS Single-Account Access - Tutorial ini di situs web Microsoft menjelaskan cara mengatur Microsoft Entra (sebelumnya dikenal sebagai Azure AD) sebagai penyedia identitas (iDP) menggunakan federasi SAMP.</p>
Centrify	<p>Konfigurasi Centrify dan Gunakan SAMP untuk SSO AWS - Halaman ini di situs web Centrify menjelaskan cara mengonfigurasi Centrify untuk menggunakan SAMP untuk SSO. AWS</p>
CyberArk	<p>Konfigurasi CyberArk untuk menyediakan Amazon Web Services (AWS) akses ke pengguna yang masuk melalui SAMP single sign-on (SSO) dari User Portal. CyberArk</p>
ForgeRock	<p>Platform ForgeRock Identitas terintegrasi dengan AWS. Anda dapat mengonfigurasi ForgeRock untuk meneruskan tag sesi. Untuk informasi lebih lanjut, lihat Kontrol Akses Berbasis Atribut untuk Amazon Web Services.</p>
Google Workspace	<p>Aplikasi cloud Amazon Web Services — Artikel di situs Bantuan Admin Google Workspace ini menjelaskan cara mengonfigurasi Google Workspace sebagai IDP SAMP 2.0 sebagai penyedia layanan. AWS</p>

Solusi	Informasi lebih lanjut
IBM	Anda dapat mengonfigurasi 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 Single Sign On (SSO) dengan Amazon AWS - Artikel ini di JumpCloud situs web 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.
Microsoft Active Directory Federation Services (AD FS)	Catatan Lapangan: 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 (IAM Identity Center). IAM Identity Center mendukung federasi identitas dengan SAMP 2.0, memungkinkan integrasi dengan solusi AD FS. Pengguna dapat masuk ke portal Pusat Identitas IAM dengan kredensi perusahaan mereka mengurangi overhead admin untuk mempertahankan kredensial terpisah di IAM Identity Center. 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 membangun akses aman AWS untuk perusahaan dan kontrol penuh atas akses AWS aplikasi.

Solusi	Informasi lebih lanjut
Okta	<p>Mengintegrasikan Antarmuka Baris Perintah Amazon Web Services Menggunakan Okta - Dari halaman ini di situs dukungan Okta, Anda dapat mempelajari cara mengonfigurasi Okta untuk digunakan. AWS Anda dapat mengonfigurasi Okta untuk meneruskan tag sesi. Untuk informasi selengkapnya, lihat Okta dan AWS Mitra untuk Menyederhanakan Akses Melalui Tag Sesi.</p>
Okta	<p>AWS Federasi Akun — Bagian ini di situs web Okta menjelaskan cara mengatur dan mengaktifkan Pusat Identitas IAM untuk AWS</p>
OneLogin	<p>Dari OneLoginKnowledgebase, 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 meneruskan tag sesi. Untuk informasi selengkapnya, lihat OneLogin dan Tag Sesi: Kontrol Akses Berbasis Atribut untuk AWS Sumber Daya.</p>
Identitas Ping	<p>PingFederate AWS Konektor - Lihat detail tentang PingFederate AWS Konektor, templat koneksi cepat untuk mengatur koneksi masuk tunggal (SSO) dan penyediaan dengan mudah. Baca dokumentasi dan unduh PingFederate AWS Konektor terbaru untuk integrasi dengan AWS. Anda dapat mengonfigurasi Identitas Ping untuk meneruskan tag sesi. Untuk informasi selengkapnya, lihat Mengumumkan Dukungan Identitas Ping untuk Kontrol Akses Berbasis Atribut di AWS.</p>
RadiantLogic	<p>Radiant Logic Technology Partners - Layanan Identitas RadiantOne Federasi Radiant Logic terintegrasi dengan AWS untuk menyediakan pusat identitas untuk SSO berbasis SAMP.</p>

Solusi	Informasi lebih lanjut
RSA	Amazon Web Services - Panduan Implementasi Siap RSA memberikan panduan untuk mengintegrasikan AWS dan RSA. Untuk informasi selengkapnya tentang konfigurasi SAMP, lihat Amazon Web Services - Konfigurasi SSO Halaman Saya SALL - Panduan Implementasi Siap RSA .
Salesforce.com	Cara mengkonfigurasi SSO dari Salesforce ke AWS - Artikel cara ini di situs pengembang Salesforce.com menjelaskan cara mengatur penyedia identitas (IdP) di Salesforce dan mengkonfigurasi sebagai penyedia layanan. AWS
SecureAuth	AWS - SecureAuth SAMP SSO - Artikel ini di SecureAuth situs web menjelaskan cara mengatur integrasi SAMP dengan AWS untuk alat. SecureAuth
Shibboleth	Cara Menggunakan Shibboleth untuk SSO ke AWS Management Console - Entri ini di Blog AWS Keamanan menyediakan step-by-step tutorial tentang cara mengatur Shibboleth dan mengkonfigurasinya sebagai penyedia identitas untuk. AWS Anda dapat mengonfigurasi Shibboleth untuk meneruskan tag sesi .

Untuk detail selengkapnya, lihat halaman [Mitra IAM](#) di AWS situs web.

Konfigurasi pernyataan SAMP untuk respons otentikasi

Setelah Anda memverifikasi identitas pengguna di organisasi Anda, penyedia identitas eksternal (IdP) mengirimkan respons otentikasi ke titik akhir AWS SAMP di `https://region-code.signin.aws.amazon.com/saml` [Untuk daftar potensi penggantian kode wilayah, lihat kolom Wilayah di titik akhir Masuk AWS](#). Respons ini adalah permintaan POST yang menyertakan token SAML yang mematuhi standar [Pengkatan HTTP POST untuk SAML 2.0](#) dan berisi elemen, atau klaim berikut. Anda mengonfigurasi klaim ini di IdP yang kompatibel dengan SAML. Lihat dokumentasi IdP Anda untuk instruksi tentang cara memasukkan klaim ini.

Ketika IDP mengirimkan respons yang berisi klaim AWS, banyak klaim yang masuk memetakan ke AWS kunci konteks. Kunci konteks ini dapat diperiksa di kebijakan IAM menggunakan elemen Condition. Daftar pemetaan yang tersedia mengikuti di bagian ini [Memetakan atribut SAMP untuk AWS mempercayai kunci konteks kebijakan](#).

Subject dan NameID

Kebijakan berikut menunjukkan sebuah contoh. Gantikan nilai Anda sendiri untuk nilai yang ditandai. Harus ada tepat satu elemen SubjectConfirmation dengan elemen SubjectConfirmationData yang menyertakan atribut NotOnOrAfter, dan atribut Recipient. Atribut ini mencakup nilai yang harus cocok dengan AWS titik akhir `https://region-code.signin.aws.amazon.com/saml`. Untuk daftar kemungkinan nilai *kode wilayah*, *lihat kolom Region* di titik akhir [AWS Masuk](#). Untuk AWS nilainya, Anda juga dapat menggunakan `https://signin.aws.amazon.com/static/saml`, seperti yang ditunjukkan pada contoh berikut.

NameID elemen dapat memiliki nilai persisten, sementara, atau terdiri dari URI Format lengkap seperti yang disediakan oleh solusi iDP. Nilai persisten menunjukkan bahwa nilai dalam NameID adalah 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

```
<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://signin.aws.amazon.com/saml"/>
```

```
</SubjectConfirmation>
</Subject>
```

Important

Kunci `saml:aud` konteks berasal dari atribut penerima SAMP karena SAMP 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 [Melewati 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 multivalued 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>
```

Role Atribut 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 yang pengguna dipetakan oleh idP Anda. [Peran IAM dan penyedia identitas IAM ditentukan sebagai pasangan ARN yang dibatasi koma dalam format yang sama dengan `RoleArn` dan `PrincipalArn` parameter yang diteruskan ke `SAMP.AssumeRoleWith`](#) 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 AWS Management Console

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 pengenalan 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 AWS Management Console. 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.doe`) atau alamat email (`john.doe@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 AWS Management Console sebelum harus meminta kredensi 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 SAMP untuk menyediakan akses masuk tunggal ke titik akhir web login AWS Management Console melalui konsol di `https://region-code.signin.aws.amazon.com/saml` Untuk daftar kemungkinan nilai *kode wilayah*, lihat kolom *Region* di titik akhir [AWS Masuk](#). Anda dapat menggunakan URL berikut secara opsional: `https://signin.aws.amazon.com/static/saml`. Perhatikan bahwa atribut ini hanya memperluas sesi ke AWS Management Console. 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 kredensial 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 pengenalan untuk orang atau aplikasi yang menggunakan peran IAM. Nilai untuk identitas sumber tetap ada saat Anda menggunakan sesi SAMP 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 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 biasanya merupakan atribut yang berhubungan dengan pengguna seperti id pengguna (johndoe) 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 dikonfigurasi 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 `DiegoRamirez` menggunakan atribut berikut.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/SourceIdentity">
  <AttributeValue>DiegoRamirez</AttributeValue>
```

Memetakan atribut SAMP untuk AWS mempercayai kunci konteks kebijakan

Tabel di bagian ini mencantumkan atribut SAML yang umum digunakan dan cara mereka dipetakan untuk mempercayai 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.

Atribut eduPerson dan eduOrg

Atribut eduPerson atau eduOrg (kunci Name)	Peta 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 PrimaryAffiliation	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 ScopedAffiliation	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)	Peta 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

Atribut Active Directory

Atribut AD	Peta 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
http://schemas.microsoft.com/ws/2008/06/identity/claims/primarygroupsid	uid	String

atribut X.500

atribut X.500	Peta 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 pengguna gabungan SAML 2.0 untuk mengakses AWS Management Console

Anda dapat menggunakan peran untuk mengonfigurasi penyedia identitas (IdP) yang sesuai dengan SAML 2.0 dan AWS untuk mengizinkan pengguna gabungan mengakses AWS Management Console. Peran ini memberikan izin kepada pengguna untuk melaksanakan tugas di konsol. Jika Anda ingin memberi pengguna gabungan SAML 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 \(Alat untuk Windows PowerShell\)](#)
- AWSAPI: [Beralih ke peran IAM \(AWS API\)](#)

Gambaran Umum

Diagram berikut menggambarkan alur untuk sistem masuk tunggal yang mendukung SAML.

Note

Penggunaan SAML khusus ini berbeda dari yang lebih umum yang diilustrasikan di [Federasi SAML 2.0](#) karena alur kerja ini membuka AWS Management Console atas nama pengguna. Ini memerlukan penggunaan endpoint AWS masuk alih-alih memanggil API secara langsung. AssumeRoleWithSAML Titik akhir memanggil API untuk pengguna dan mengembalikan URL yang secara otomatis mengalihkan peramban pengguna ke AWS Management Console.

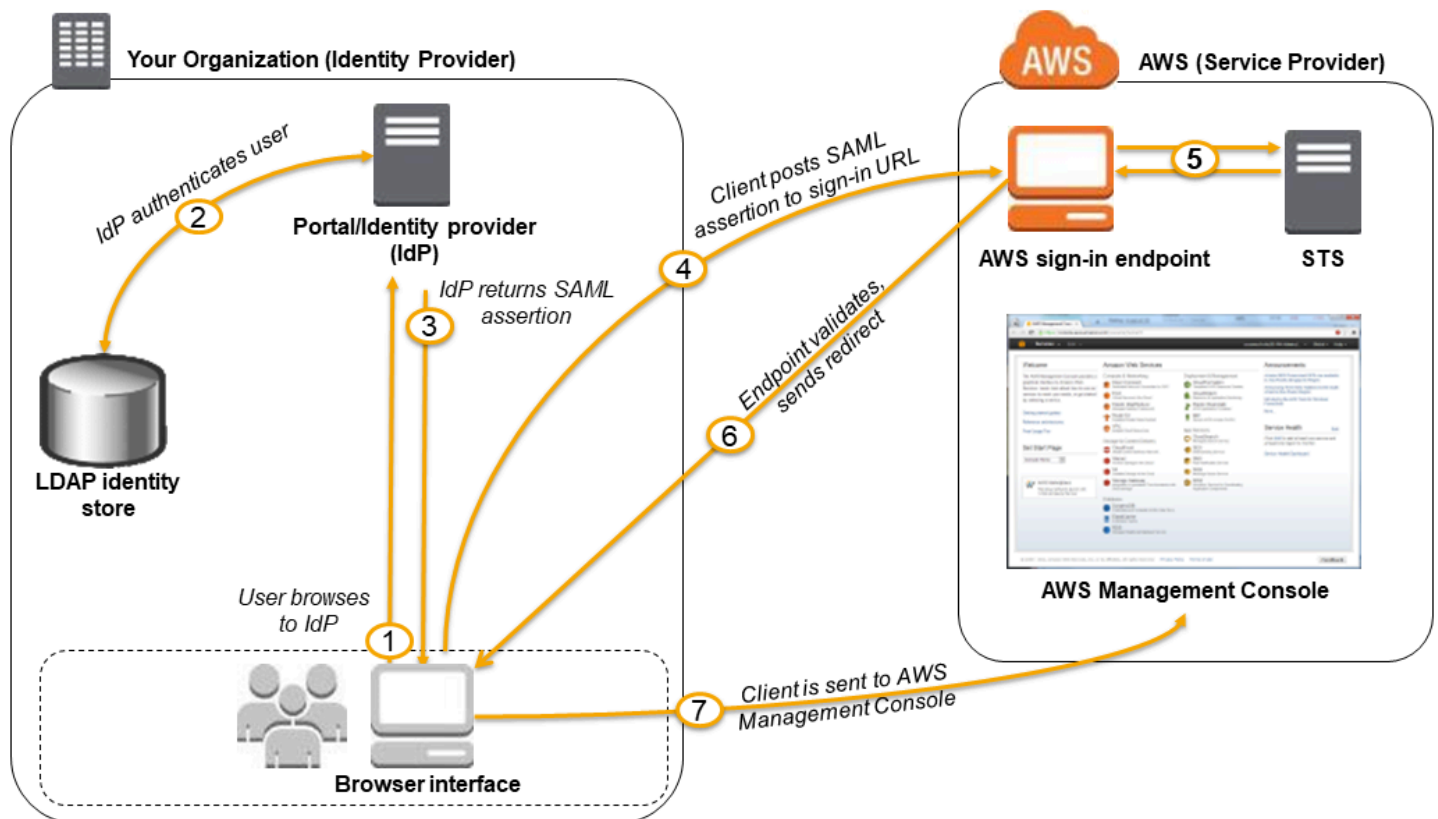


Diagram ini menggambarkan langkah-langkah berikut:

1. Pengguna menelusuri portal organisasi Anda dan memilih opsi untuk membuka AWS Management Console. Di organisasi, portal biasanya merupakan fungsi IdP Anda yang menangani pertukaran kepercayaan antara organisasi Anda dan 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. Peramban klien dialihkan ke titik akhir masuk tunggal AWS dan memposting pernyataan SAML.
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 AWS Management Console. Jika tanggapan autentikasi SAML mencakup atribut yang memetakan ke beberapa peran IAM, pengguna pertama-tama diminta memilih peran untuk mengakses konsol.

Dari perspektif pengguna, proses terjadi secara transparan: Pengguna memulai di portal internal organisasi Anda dan berakhir di AWS Management Console, tanpa harus menyediakan kredensial AWS.

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 mengonfigurasi penyimpanan identitas (seperti Windows Active Directory) untuk bekerja dengan IdP berbasis SAML 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 guna merutekan permintaan pengguna untuk AWS Management Console ke titik akhir SAML AWS untuk autentikasi menggunakan pernyataan SAML. Cara Anda mengonfigurasi IdP untuk menghasilkan file metadata.xml tergantung pada IdP Anda. Lihat dokumentasi IdP untuk instruksi, atau lihat [Integrasikan penyedia solusi SAMP pihak ketiga dengan AWS](#) untuk tautan ke dokumentasi web bagi banyak penyedia SAML yang didukung.

Membuat penyedia SAML di IAM

Berikutnya, masuk ke AWS Management Console dan buka konsol IAM. Di sana Anda membuat penyedia SAML baru, yang merupakan entitas dalam 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 SAMP di IAM](#).

Konfigurasi izin di AWS untuk pengguna gabungan Anda

Langkah selanjutnya adalah membuat 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 tersebut juga menentukan apa yang diizinkan untuk dilakukan oleh pengguna yang diautentikasi oleh IdP organisasi Anda di 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 AWS Management Console. Kebijakan kepercayaan untuk peran tersebut mungkin akan terlihat seperti ini:

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Principal": {"Federated": "arn:aws:iam::account-id:saml-provider/ExampleOrgSSOProvider"},
    "Action": "sts:AssumeRoleWithSAML",
    "Condition": {"StringEquals": {
      "saml:edupersonorgdn": "ExampleOrg",
      "saml:aud": "https://signin.aws.amazon.com/saml"
    }}
  ]
}
```

Note

IDP SALL yang digunakan dalam kebijakan kepercayaan peran harus berada dalam akun yang sama dengan peran tersebut.

Anda dapat menyertakan endpoint regional untuk `saml:aud` atribut di `https://region-code.signin.aws.amazon.com/static/saml-metadata.xml`. Untuk daftar kemungkinan nilai *kode wilayah*, lihat kolom Region di titik akhir [AWSMasuk](#).

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 SAMP 2.0 \(konsol\)](#).

Selesaikan konfigurasi dan buat pernyataan SAML

Beri tahu SALL 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` Untuk daftar kemungkinan nilai *kode wilayah*, lihat kolom Region di titik akhir [AWSMasuk](#).

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 SAMP 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 di AWS sebagai kunci konteks syarat yang dapat Anda evaluasi dalam kebijakan. 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. Untuk detailnya, lihat [Konfigurasi pernyataan SAMP untuk respons otentikasi](#).

Kredensial keamanan sementara di IAM

Anda dapat menggunakan AWS Security Token Service (AWS STS) untuk membuat dan menyediakan kredensial keamanan sementara kepada pengguna tepercaya yang dapat mengontrol akses ke sumber daya Anda AWS . 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 sebagai berikut dibandingkan kredensial jangka panjang:

- Anda tidak perlu mendistribusikan atau menanamkan kredensial 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](#).
- Kredensial 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.

AWS STS dan 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 [Mengelola AWS STS dalam 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 informasi lebih lanjut tentang penyedia identitas eksternal, lihat [Penyedia dan federasi identitas](#).

- Federasi SAFL — Anda dapat mengautentikasi 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 kredensial masuk yang berbeda. Ini dikenal sebagai pendekatan masuk tunggal untuk akses sementara. AWS STS mendukung standar terbuka seperti Security Assertion Markup Language (SAFL) 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](#).
- Pialang federasi khusus — Anda dapat menggunakan sistem otentikasi organisasi Anda untuk memberikan akses ke AWS sumber daya. Untuk contoh skenario, lihat [Mengaktifkan akses broker identitas khusus ke konsol AWS](#).
- Federasi menggunakan SAFL 2.0 — Anda dapat menggunakan sistem otentikasi organisasi Anda dan SALL untuk memberikan akses ke sumber daya. AWS Untuk informasi lebih lanjut dan contoh skenario, lihat [Federasi SAML 2.0](#).
- Federasi OpenID Connect (OIDC) - Anda dapat mengizinkan pengguna masuk menggunakan penyedia identitas pihak ketiga yang terkenal seperti Login with Amazon, Facebook, Google, atau penyedia yang kompatibel dengan OIDC 2.0 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 Akun AWS keamanan Anda, karena Anda tidak perlu mendistribusikan kredensial keamanan jangka panjang, seperti kunci akses pengguna IAM, dengan aplikasi Anda. Untuk informasi selengkapnya, lihat [Federasi OIDC](#).

AWS STS Federasi OIDC mendukung Login with 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 mengautentikasi mereka untuk akses aman ke sumber daya Anda. AWS Amazon Cognito mendukung penyedia identitas yang sama dengan AWS STS, dan juga mendukung akses (tamu) yang tidak diautentikasi 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 [Otentikasi dengan Amplify di Dokumentasi Amplify](#).

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 [Membuat peran untuk mendelegasikan 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 kredensial keamanan sementara ke instans 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 pada aplikasi yang berjalan di instans Amazon EC2..](#)

AWS Layanan lainnya

Anda dapat menggunakan kredensial keamanan sementara untuk mengakses sebagian besar AWS layanan. Untuk daftar layanan yang menerima kredensial keamanan sementara, lihat [AWS layanan yang bekerja dengan IAM](#).

Meminta kredensial keamanan sementara

Untuk meminta kredensial keamanan sementara, Anda dapat menggunakan operasi AWS Security Token Service (AWS STS) di API. AWS Ini termasuk operasi untuk membuat dan menyediakan kredensial keamanan sementara kepada pengguna tepercaya yang dapat mengontrol akses ke sumber daya Anda AWS . Untuk informasi lebih lanjut tentang AWS STS, lihat [Kredensial 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 [Menggunakan peran IAM](#).

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: [AWS Command Line Interface](#), dan [AWS Tools for Windows PowerShell](#).

Operasi AWS STS API membuat sesi baru dengan kredensial keamanan sementara yang mencakup access key pair dan session token. 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 [Melewati tag sesi di AWS STS](#).

Note

Ukuran token sesi yang dikembalikan operasi AWS STS 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.

Menggunakan 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 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 [Mengelola AWS STS dalam Wilayah AWS](#).

Berikut ini adalah operasi API yang dapat Anda gunakan untuk memperoleh kredensial sementara untuk digunakan di AWS lingkungan dan aplikasi Anda.

[AssumeRole](#) pendelegasian lintas akun dan federasi melalui broker identitas khusus

Operasi `AssumeRole` 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 tempat 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 kredensi aktif. Untuk mempelajari siapa yang dapat memanggil operasi ini, lihat [Membandingkan operasi AWS STS API](#). Lihat informasi yang lebih lengkap di [Membuat peran untuk mendelegasikan izin kepada pengguna IAM](#) dan [Mengonfigurasi akses API yang dilindungi MFA](#).

Panggilan ini harus dilakukan menggunakan kredensial AWS keamanan yang valid. Saat Anda melakukan panggilan ini, Anda menyampaikan informasi berikut:

- Amazon Resource Name (ARN) dari peran yang harus diasumsikan oleh aplikasi.
- (Opsional) Durasi, yang menentukan durasi kredensial keamanan sementara. Gunakan parameter `DurationSeconds` untuk menentukan durasi sesi peran dari 900 detik (15 menit) hingga pengaturan durasi sesi maksimum untuk peran tersebut. Untuk mempelajari cara melihat nilai maksimum untuk peran Anda, lihat [Lihat pengaturan durasi sesi maksimum untuk peran](#). Jika Anda tidak meneruskan parameter ini, kredensial sementara akan kedaluwarsa dalam satu jam. Parameter `DurationSeconds` dari API ini terpisah dari Parameter HTTP `SessionDuration` yang Anda gunakan untuk menentukan durasi sesi konsol. Gunakan Parameter HTTP `SessionDuration` dalam permintaan ke titik akhir federasi untuk token masuk konsol. Untuk informasi selengkapnya, lihat [Mengaktifkan akses broker identitas khusus ke konsol AWS](#).
- Nama sesi peran. Gunakan nilai string ini untuk mengidentifikasi sesi ketika peran digunakan oleh penanggung jawab yang berbeda. Untuk tujuan keamanan, administrator dapat melihat bidang ini dalam [log AWS CloudTrail](#) untuk membantu mengidentifikasi siapa yang melakukan tindakan di

AWS. Administrator Anda mungkin mengharuskan Anda untuk menentukan nama pengguna IAM sebagai nama sesi saat Anda mengasumsikan peran tersebut. Untuk informasi selengkapnya, lihat [sts:RoleSessionName](#).

- (Opsional) Sumber identitas. Anda dapat meminta pengguna untuk menentukan identitas sumber ketika mereka mengambil peran. Setelah identitas sumber diatur, nilai tidak dapat diubah. Hal ini terdapat dalam permintaan untuk semua tindakan yang diambil selama sesi peran. Nilai identitas sumber tetap diseluruh sesi [peran berantai](#). Anda dapat menggunakan informasi identitas sumber dalam AWS CloudTrail log untuk menentukan siapa yang mengambil tindakan dengan peran. Untuk informasi selengkapnya tentang penggunaan identitas sumber, lihat [Memantau dan mengontrol tindakan yang diambil dengan peran yang diasumsikan](#).
- (Opsional) Kebijakan sesi inline atau terkelola. Kebijakan ini membatasi izin dari kebijakan berbasis identitas peran yang ditetapkan untuk sesi peran. 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 ini, lihat [Kebijakan sesi](#).
- (Opsional) Tanda sesi. Anda dapat mengambil peran dan kemudian menggunakan kredensial sementara untuk membuat permintaan. Saat Anda melakukannya, tanda penanggung jawab sesi ini mencakup tanda peran dan tanda sesi yang diteruskan. Jika Anda melakukan panggilan ini menggunakan kredensial sementara, sesi baru juga akan menurunkan tanda sesi transitif dari sesi panggilan. Untuk informasi lebih lanjut tentang tanda sesi, lihat [Melewati tag sesi di AWS STS](#).
- (Opsional) Informasi MFA. Jika dikonfigurasi untuk menggunakan Autentikasi Multi-Faktor (MFA) maka Anda menyertakan pengidentifikasi untuk perangkat MFA dan kode sekali pakai yang disediakan oleh perangkat tersebut.
- (Opsional) Nilai ExternalId yang dapat digunakan saat mendelegasikan akses akun Anda ke pihak ketiga. Nilai ini membantu memastikan bahwa hanya pihak ketiga yang ditentukan yang dapat mengakses peran tersebut. Untuk informasi selengkapnya, lihat [Cara menggunakan ID eksternal saat memberikan akses ke sumber daya AWS Anda kepada pihak ketiga](#).

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%22012-10-17%22%2C%22Statement%22%3A%5B%7B%22Sid%22%3A%20%22Stmnt1%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 ditunjukkan dalam contoh sebelumnya adalah versi kebijakan berkode URL dari kebijakan berikut ini:

```
{"Version":"2012-10-17","Statement":  
[{"Sid":"Stmnt1","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 AWS HTTP API. 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 AWS Permintaan Dengan Menggunakan Tanda Tangan Versi 4](#) di bagian 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//////////wEXAMPLEtc764bNrC9SAPBSM22wD0k4x4HIZ8j4FZTwdQW
    LWSKWHGBuFqwAeMicRXmxfpSPfIeoIYRqTf1fKD8YUuwthAx7mSEI/qkPpKPi/kMcGd
    QrmGdeehM4IC1NtBmUpp2wUE8phUZampKsburEDy0KPkYQDYwT7WZ0wq5VSXDvp75YU
    9HFv1Rd8Tx6q6fE8YQcHNvXAKiY9q6d+xo0rKwT38xVqr7ZD0u0iPPkUL64lIZbqBAz
    +scqKmlzm8FDrypNC9Yjc8fP0Ln9FX9KSYvKTr4rvx3iSI1TJabIQwj2ICCR/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>AR0123EXAMPLE123:John</AssumedRoleId>
</AssumedRoleUser>
<PackedPolicySize>8</PackedPolicySize>
</AssumeRoleResult>
<ResponseMetadata>
  <RequestId>c6104cbe-af31-11e0-8154-cbc7ccf896c7</RequestId>
</ResponseMetadata>
</AssumeRoleResponse>

```

Note

AWS Konversi memampatkan kebijakan sesi dan tag sesi yang diteruskan ke dalam format biner yang dikemas yang memiliki batas terpisah. Permintaan Anda bisa gagal untuk batas ini bahkan jika teks biasa Anda memenuhi persyaratan lain. Elemen respons `PackedPolicySize` menunjukkan persentase seberapa dekat kebijakan dan tanda untuk permintaan Anda dengan batas ukuran atas.

[AssumeRoleWithWebIdentity](#)—federasi melalui penyedia identitas berbasis web

`AssumeRoleWithWebIdentity` Operasi API mengembalikan satu set kredensial keamanan sementara untuk pengguna federasi yang diautentikasi melalui penyedia identitas umum. Contoh penyedia identitas umum mencakup Login with Amazon, Facebook, Google, atau penyedia identitas mana pun yang kompatibel dengan OpenID Connect (OIDC). Operasi ini berguna untuk membuat aplikasi mobile atau aplikasi web berbasis klien yang memerlukan akses ke . AWS Menggunakan

operasi ini berarti bahwa pengguna Anda tidak memerlukan identitas mereka sendiri AWS atau IAM. Untuk informasi selengkapnya, lihat [Federasi OIDC](#).

Alih-alih menelepon langsung `AssumeRoleWithWebIdentity`, kami menyarankan Anda menggunakan Amazon Cognito dan penyedia kredensi Amazon Cognito dengan AWS SDK untuk pengembangan seluler. Untuk informasi selengkapnya, lihat [Otentikasi dengan Amplify di Dokumentasi Amplify](#).

Jika Anda tidak menggunakan Amazon Cognito, Anda menyebut `AssumeRoleWithWebIdentity` tindakan. AWS STS Ini adalah panggilan tanpa tanda tangan, yang berarti bahwa aplikasi tidak perlu memiliki akses ke kredensial keamanan AWS untuk membuat panggilan. Saat Anda melakukan panggilan ini, Anda menyampaikan informasi berikut:

- Amazon Resource Name (ARN) dari peran yang harus diasumsikan oleh aplikasi. Jika aplikasi Anda mendukung beberapa cara bagi pengguna untuk masuk, Anda harus menentukan beberapa peran, satu per penyedia identitas. Panggilan ke `AssumeRoleWithWebIdentity` harus menyertakan ARN peran yang spesifik untuk penyedia yang digunakan pengguna untuk masuk.
- Token yang diperoleh aplikasi dari IdP setelah aplikasi mengautentikasi pengguna.
- Anda dapat mengonfigurasi IdP Anda untuk meneruskan atribut ke dalam token Anda sebagai [tanda sesi](#).
- (Opsional) Durasi, yang menentukan durasi kredensial keamanan sementara. Gunakan parameter `DurationSeconds` untuk menentukan durasi sesi peran dari 900 detik (15 menit) hingga pengaturan durasi sesi maksimum untuk peran tersebut. Untuk mempelajari cara melihat nilai maksimum untuk peran Anda, lihat [Lihat pengaturan durasi sesi maksimum untuk peran](#). Jika Anda tidak meneruskan parameter ini, kredensial sementara akan kedaluwarsa dalam satu jam. Parameter `DurationSeconds` dari API ini terpisah dari Parameter HTTP `SessionDuration` yang Anda gunakan untuk menentukan durasi sesi konsol. Gunakan Parameter HTTP `SessionDuration` dalam permintaan ke titik akhir federasi untuk token masuk konsol. Untuk informasi selengkapnya, lihat [Mengaktifkan akses broker identitas khusus ke konsol AWS](#).
- Nama sesi peran. Gunakan nilai string ini untuk mengidentifikasi sesi ketika peran digunakan oleh penanggung jawab yang berbeda. Untuk tujuan keamanan, administrator dapat melihat bidang ini dalam [log AWS CloudTrail](#) untuk mempelajari siapa yang melakukan tindakan di AWS. Administrator Anda mungkin akan meminta Anda memberikan nilai spesifik untuk nama sesi ketika Anda mengasumsikan peran tersebut. Untuk informasi selengkapnya, lihat [sts:RoleSessionName](#).
- (Opsional) Sumber identitas. Anda dapat meminta pengguna gabungan untuk menentukan identitas sumber saat mereka mengambil peran. Setelah identitas sumber diatur, nilai tidak

dapat diubah. Hal ini terdapat dalam permintaan untuk semua tindakan yang diambil selama sesi peran. Nilai identitas sumber tetap diseluruh sesi [peran berantai](#). Anda dapat menggunakan informasi identitas sumber dalam AWS CloudTrail log untuk menentukan siapa yang mengambil tindakan dengan peran. Untuk informasi selengkapnya tentang penggunaan identitas sumber, lihat [Memantau dan mengontrol tindakan yang diambil dengan peran yang diasumsikan](#).

- (Opsional) Kebijakan sesi inline atau terkelola. Kebijakan ini membatasi izin dari kebijakan berbasis identitas peran yang ditetapkan untuk sesi peran. 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 ini, lihat [Kebijakan sesi](#).

Note

Sebuah panggilan ke `AssumeRoleWithWebIdentity` 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.

Saat Anda menelepon `AssumeRoleWithWebIdentity` AWS, verifikasi keaslian token. Misalnya, tergantung pada penyedia, AWS mungkin melakukan panggilan ke penyedia dan menyertakan token yang telah dilewati aplikasi. Dengan asumsi bahwa penyedia identitas memvalidasi token, 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.


Bila Anda memiliki kredensial keamanan sementara, Anda dapat menggunakannya untuk melakukan panggilan AWS API. Ini adalah proses yang sama dengan 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. Sebagaimana diketahui, secara default kredensial kedaluwarsa setelah satu jam. Jika Anda tidak menggunakan `CredentialsProvider`

operasi [Amazonsts](#) di AWS SDK, terserah Anda dan aplikasi Anda untuk menelepon lagi. `AssumeRoleWithWebIdentity` Panggil operasi ini untuk mendapatkan rangkaian kredensial keamanan sementara yang baru sebelum kredensial lama kedaluwarsa.

[AssumeRoleWithSAML—federasi](#) melalui Penyedia Identitas perusahaan yang kompatibel dengan SAFL 2.0

Operasi API `AssumeRoleWithSAML` mengembalikan serangkaian kredensial keamanan sementara untuk pengguna federasi yang diautentikasi oleh sistem identitas yang ada milik organisasi Anda. 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](#).

 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.

Ini adalah panggilan tanpa tanda tangan, yang berarti bahwa aplikasi tidak perlu memiliki akses ke kredensial keamanan AWS untuk membuat panggilan. Saat Anda melakukan panggilan ini, Anda menyampaikan informasi berikut:

- Amazon Resource Name (ARN) dari peran yang harus diasumsikan oleh aplikasi.
- ARN dari penyedia SAML yang dibuat di IAM yang menjelaskan penyedia identitas.
- Pernyataan SAML, yang dikodekan di basis64, yang disediakan oleh penyedia identitas SAML dalam respon autentikasinya terhadap permintaan masuk dari aplikasi Anda.
- Anda dapat mengonfigurasi Anda untuk meneruskan atribut ke pernyataan SAML Anda sebagai [tanda sesi](#).
- (Opsional) Durasi, yang menentukan durasi kredensial keamanan sementara. Gunakan parameter `DurationSeconds` untuk menentukan durasi sesi peran dari 900 detik (15 menit) hingga pengaturan durasi sesi maksimum untuk peran tersebut. Untuk mempelajari cara melihat nilai

maksimum untuk peran Anda, lihat [Lihat pengaturan durasi sesi maksimum untuk peran](#). Jika Anda tidak meneruskan parameter ini, kredensial sementara akan kedaluwarsa dalam satu jam. Parameter `DurationSeconds` dari API ini terpisah dari Parameter `HTTP SessionDuration` yang Anda gunakan untuk menentukan durasi sesi konsol. Gunakan Parameter `HTTP SessionDuration` dalam permintaan ke titik akhir federasi untuk token masuk konsol. Untuk informasi selengkapnya, lihat [Mengaktifkan akses broker identitas khusus ke konsol AWS](#).

- (Opsional) Kebijakan sesi inline atau terkelola. Kebijakan ini membatasi izin dari kebijakan berbasis identitas peran yang ditetapkan untuk sesi peran. 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 ini, lihat [Kebijakan sesi](#).
- Nama sesi peran. Gunakan nilai string ini untuk mengidentifikasi sesi ketika peran digunakan oleh penanggung jawab yang berbeda. Untuk tujuan keamanan, administrator dapat melihat bidang ini dalam [log AWS CloudTrail](#) untuk mempelajari siapa yang melakukan tindakan di AWS. Administrator Anda mungkin akan meminta Anda memberikan nilai spesifik untuk nama sesi ketika Anda mengasumsikan peran tersebut. Untuk informasi selengkapnya, lihat [sts:RoleSessionName](#).
- (Opsional) Sumber identitas. Anda dapat meminta pengguna gabungan untuk menentukan identitas sumber saat mereka mengambil peran. Setelah identitas sumber diatur, nilai tidak dapat diubah. Hal ini terdapat dalam permintaan untuk semua tindakan yang diambil selama sesi peran. Nilai identitas sumber tetap diseluruh sesi [peran berantai](#). Anda dapat menggunakan informasi identitas sumber dalam AWS CloudTrail log untuk menentukan siapa yang mengambil tindakan dengan peran. Untuk informasi selengkapnya tentang penggunaan identitas sumber, lihat [Memantau dan mengontrol tindakan yang diambil dengan peran yang diasumsikan](#).

Saat Anda menelepon `AssumeRoleWithSAML`, AWS verifikasi 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 SAFL. Saat digabungkan dengan elemen `Subject`, mereka dapat mengidentifikasi pengguna federasi secara unik.
- 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 [Konfigurasi pernyataan SAMP untuk respons otentikasi](#).

Bila Anda memiliki kredensial keamanan sementara, Anda dapat menggunakannya untuk melakukan panggilan AWS API. Ini adalah proses yang sama dengan 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 `CredentialsProvider` tindakan [Amazonsts](#) di AWS SDK, terserah Anda dan aplikasi Anda untuk menelepon lagi. `AssumeRoleWithSAML` Hubungi operasi ini untuk mendapatkan set keamanan kredensial sementara yang baru sebelum kredensial lama kedaluwarsa.

[GetFederationToken](#)—federasi melalui pialang identitas khusus

Operasi API `GetFederationToken` mengembalikan satu set kredensial keamanan sementara untuk 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. Kredensi yang dihasilkan berlaku untuk durasi yang ditentukan, antara 900 detik (15 menit) hingga 129.600 detik (36 jam). Masa kedaluwarsa yang lebih lama dapat membantu mengurangi jumlah panggilan AWS karena Anda tidak perlu mendapatkan kredensi baru sesering mungkin.

Saat Anda membuat permintaan ini, Anda menggunakan kredensial pengguna IAM tertentu. Izin untuk kredensial keamanan sementara ditentukan oleh kebijakan sesi yang Anda lewatkan saat Anda memanggil `GetFederationToken`. Izin sesi yang dihasilkan adalah persimpangan kebijakan pengguna IAM dan kebijakan sesi yang Anda lewatkan. Kebijakan sesi tidak dapat digunakan untuk memberikan lebih banyak izin daripada yang diizinkan oleh kebijakan berbasis-identitas dari

pengguna IAM yang meminta federasi. Untuk informasi lebih lanjut tentang izin sesi peran ini, lihat [Kebijakan sesi](#).

Saat Anda menggunakan kredensial sementara yang dikembalikan oleh operasi `GetFederationToken`, tanda utama sesi mencakup tanda pengguna dan tanda sesi yang lulus. Untuk informasi lebih lanjut tentang tanda sesi, lihat [Melewati tag sesi di AWS STS](#).

`GetFederationTokenPanggilan` mengembalikan kredensial 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](#) dan [tanda 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 dalam contoh sebelumnya mencakup ARN berkode URL 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 AWS HTTP API. 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, buka [AWS](#)

[Permintaan Menandatangani Dengan Menggunakan Tanda Tangan Versi 4](#) di bagian 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//////////wEXAMPLEtc764bNrC9SAPBSM22wD0k4x4HIZ8j4FZTwdQW
    LWSKWHGBuFqwAeMicRXmxfpSPfIeoIYRqTf1fKD8YUuwthAx7mSEI/qkPpKPi/kMcGd
    QrmGdeehM4IC1NtBmUpp2wUE8phUZampKsburEDy0KPKyQDYwT7WZ0wq5VSXDvp75YU
    9HFv1Rd8Tx6q6fE8YQcHNvXAKiY9q6d+xo0rKwT38xVqr7ZD0u0iPPkUL64lIZbqBAz
    +scqKmlzm8FDrypNC9Yjc8fPOLn9FX9KSYvKTr4rvx3iSI1TJabIQwj2ICCEXAMPLE==
  </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>4</PackedPolicySize>
</GetFederationTokenResult>
<ResponseMetadata>
<RequestId>c6104cbe-af31-11e0-8154-cbc7ccf896c7</RequestId>
</ResponseMetadata>
</GetFederationTokenResponse>
```

Note

AWS Konversi memampatkan kebijakan sesi dan tag sesi yang diteruskan ke dalam format biner yang dikemas yang memiliki batas terpisah. Permintaan Anda bisa gagal untuk batas ini bahkan jika teks biasa Anda memenuhi persyaratan lain. Elemen respons

`PackedPolicySize` menunjukkan persentase seberapa dekat kebijakan dan tanda untuk permintaan Anda dengan batas ukuran atas.

AWS menyarankan agar Anda memberikan izin di tingkat sumber daya (misalnya, Anda melampirkan kebijakan berbasis sumber daya ke bucket Amazon S3), Anda dapat menghilangkan parameternya. `Policy` Namun, jika Anda tidak menyertakan kebijakan untuk pengguna federasi, kredensial 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, anggap Akun AWS nomor Anda 111122223333, dan Anda memiliki bucket Amazon S3 yang ingin Anda izinkan untuk diakses oleh 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`.

[GetSessionToken](#)—kredensial sementara 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. Untuk informasi selengkapnya, lihat [Meminta kredensial keamanan sementara](#) atau [GetSessionToken](#) di Referensi AWS Security Token Service API.

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 sebuah Pengguna root akun AWS dibatasi hingga durasi satu jam.

`GetSessionToken` mengembalikan kredensial keamanan sementara yang terdiri dari token sesi, ID kunci akses, dan kunci akses rahasia. 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 AWS HTTP API. 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, buka [AWS Permintaan Menandatangani Dengan Menggunakan Tanda Tangan Versi 4](#) di bagian 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>
        AQoEXAMPLEH4aoAH0gNCAPyJxz4B1CFFxWNE10PTgk5TthT+FvwqnKwRc0IfRrh3c/L
        To6UDdyJw00vEVPvLXCrrrUtdnniCEXAMPLE/IvU1dYUg2RVAJBanLiHb4IgrmpRV3z
        rkuWJ0gQs8IZZaIv2BXIa2R40lglkBN9bkUDNCJiBeb/AX1zBBko7b15fjrBs2+cTQtp
        Z3CYWFXG8C5zqx37wn0E49mRl/+0tkIKG07fAE
      </SessionToken>
      <SecretAccessKey>
        wJalrXUtnFEMI/K7MDENG/bPxrFiCYzEXAMPLEKEY
      </SecretAccessKey>
      <Expiration>2011-07-11T19:55:29.611Z</Expiration>
      <AccessKeyId>AKIAIOSFODNN7EXAMPLE</AccessKeyId>
    </Credentials>
  </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 kredensial keamanan sementara yang mencakup status otentikasi MFA. Kredensial keamanan sementara kemudian dapat digunakan untuk mengakses operasi API atau AWS situs web yang dilindungi MFA 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 untuk AWS STS dapat ke titik akhir global atau ke salah satu titik akhir Regional yang Anda 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 AWS HTTP API. 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 AWS Permintaan Dengan Menggunakan Tanda Tangan Versi 4](#) di bagian Referensi Umum Amazon Web Services untuk mempelajari cara menandatangani permintaan.

Membandingkan operasi AWS STS API

Tabel berikut membandingkan fitur operasi API AWS STS yang mengembalikan kredensial keamanan sementara tersebut. Untuk mempelajari tentang berbagai metode yang dapat Anda gunakan untuk meminta kredensial keamanan sementara dengan mengambil peran, lihat [Menggunakan peran IAM](#). Untuk mempelajari berbagai operasi AWS STS API yang memungkinkan Anda meneruskan tag sesi, lihat [Melewati tag sesi di AWS STS](#).

Membandingkan opsi API Anda

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
AssumeRole	Pengguna IAM atau peran IAM dengan kredensial keamanan sementara yang ada	15 m Pengaturan durasi sesi maksimum 1 jam	Ya	Ya	Tidak bisa menelepon <code>GetFederationToken</code> atau <code>GetSessionToken</code> .
AssumeRoleWithSAML	Setiap pengguna; penelepon harus melewati tanggapan autentikasi SAML yang menunjukkan autentikasi dari penyedia layanan yang dikenal	15 m Pengaturan durasi sesi maksimum 1 jam	Tidak	Ya	Tidak bisa menelepon <code>GetFederationToken</code> atau <code>GetSessionToken</code> .
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 <code>GetFederationToken</code> atau <code>GetSessionToken</code> .

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
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 <code>GetCallerIdentity</code> . SSO ke konsol diperbolehkan. ⁵
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 <code>AssumeRole</code> atau <code>GetCallerIdentity</code> . 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 [Mengonfigurasi akses API yang dilindungi MFA](#).

² Dukungan kebijakan sesi. Kebijakan sesi adalah kebijakan yang Anda jalankan sebagai parameter saat Anda secara terprogram membuat sesi sementara untuk peran atau pengguna federasi. 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 [Lihat pengaturan 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 gabungan SAML 2.0 untuk mengakses AWS Management Console](#) dan [Cara Mengaktifkan Akses Lintas Akun ke Konsol AWS Manajemen](#) di Blog AWS Keamanan.

Setelah Anda mengambil kredensi sementara Anda, Anda tidak dapat mengakses AWS Management Console dengan meneruskan kredensialnya ke titik akhir masuk tunggal federasi. Untuk informasi selengkapnya, lihat [Mengaktifkan akses broker identitas khusus ke konsol AWS](#).

Menggunakan kredensial sementara dengan sumber daya AWS.

Anda dapat menggunakan kredensial keamanan sementara untuk membuat permintaan terprogram untuk sumber daya AWS menggunakan AWS CLI atau API AWS (menggunakan [SDK AWS](#)).

Kredensi 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 di sepanjang kredensial sementara tersebut. AWS menggunakan token sesi untuk memvalidasi kredensial keamanan sementara.
- Kredensi sementara kedaluwarsa setelah interval yang ditentukan. Setelah kredensi sementara kedaluwarsa, panggilan apa pun yang Anda lakukan dengan kredensi tersebut akan gagal, jadi Anda harus membuat kumpulan kredensi sementara yang baru. Kredensi sementara tidak dapat diperpanjang atau disegarkan di luar interval yang ditentukan asli.
- 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 [Melewati tag sesi di AWS STS](#).

Jika Anda menggunakan [AWSSDK](#), [AWS Command Line Interface](#)(AWS CLI), atau [Tools untuk Windows PowerShell](#), [cara untuk](#) mendapatkan dan menggunakan kredensial keamanan sementara berbeda dengan konteksnya. Jika Anda menjalankan kode, AWS CLI, atau PowerShell perintah 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 menyediakan kredensial keamanan sementara yang dapat mengontrol akses ke sumber daya AWS Anda. Untuk informasi lebih lanjut tentang AWS STS, lihat [Kredensial 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 N.), meskipun kredensi yang Anda dapatkan dari titik 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 API AWS STS 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 [Mengelola AWS STS dalam Wilayah AWS](#).

Daftar Isi

- [Menggunakan kredensial sementara di instans Amazon EC2](#)
- [Menggunakan kredensial keamanan sementara dengan SDK AWS](#)
- [Menggunakan kredensial keamanan sementara dengan AWS CLI](#)
- [Menggunakan kredensial keamanan sementara dengan operasi API](#)
- [Informasi lain](#)

Menggunakan kredensial sementara di instans Amazon EC2

Jika Anda ingin menjalankan AWS CLI perintah atau kode di dalam instans EC2, cara yang disarankan untuk mendapatkan kredensi adalah dengan menggunakan [peran untuk Amazon EC2](#). 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.

Aplikasi, AWS CLI, dan Alat untuk PowerShell perintah Windows yang berjalan pada instance kemudian bisa mendapatkan kredensial keamanan sementara otomatis dari metadata instance. Anda tidak perlu secara eksplisit mendapatkan kredensial kerahasiaan keamanan sementara. AWSSDK, AWS CLI, dan Tools untuk Windows PowerShell secara otomatis mendapatkan kredensi 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 AWS Sumber Daya di Amazon Elastic Compute Cloud](#) - AWS SDK for Java
- [Memberikan Akses Menggunakan Peran IAM](#) - AWS SDK for .NET
- [Menciptakan Peran](#) - AWS SDK for Ruby

Menggunakan kredensial keamanan sementara dengan SDK AWS

Untuk menggunakan kredensial keamanan sementara dalam kode, Anda secara terprogram memanggil API AWS STS seperti `AssumeRole` dan mengekstraksi kredensial yang dihasilkan serta token sesi. Kemudian Anda menggunakan nilai tersebut sebagai kredensial untuk panggilan selanjutnya ke AWS. Contoh berikut ini menunjukkan pseudocode tentang cara menggunakan kredensial keamanan sementara jika Anda menggunakan SDK AWS:

```
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 \(Boto\)](#)), lihat [Beralih 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 `AssumeRoleGetFederationToken`, dan operasi API lainnya, lihat [Referensi AWS Security Token Service 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 SDK AWS di bagian [Halaman dokumentasi AWS](#), di bagian SDK dan Toolkit.

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, parameter `profile` diasumsikan sebagai profil dalam file konfigurasi file AWS CLI. 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, AWS CLI mencari kredensial dalam urutan tertentu —pertama dalam variabel lingkungan dan kemudian di file konfigurasi. Oleh karena itu, setelah Anda memasukkan kredensial sementara ke dalam variabel lingkungan, AWS CLI secara default menggunakan kredensial tersebut. (Jika Anda menentukan parameter `profile` dalam perintah, AWS CLI melompati variabel lingkungan. Sebaliknya, AWS CLI mencari di file konfigurasi, yang memungkinkan Anda untuk menimpa kredensial di variabel lingkungan jika Anda memerlukannya.)

Contoh berikut menunjukkan bagaimana Anda dapat mengatur variabel lingkungan untuk kredensial keamanan sementara dan kemudian memanggil perintah AWS CLI. Karena tidak ada parameter `profile` yang disertakan dalam perintah AWS CLI, AWS CLI mencari kredensial terlebih dahulu dalam variabel lingkungan dan oleh karena itu menggunakan kredensial sementara.

Linux

```
$ export AWS_ACCESS_KEY_ID=ASIAIOSFODNN7EXAMPLE
$ export AWS_SECRET_ACCESS_KEY=wJa1rXUtnFEMI/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=wJa1rXUtnFEMI/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 kredensial keamanan sementara yang Anda dapatkan dari AWS Security Token Service (AWS STS). Untuk melakukannya, Anda menggunakan access key ID dan secret access key yang Anda terima dari 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 ke permintaan API Anda token sesi yang Anda terima dari 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 atau parameter string kueri, tetapi tidak keduanya. Untuk informasi lebih lanjut tentang menandatangani permintaan API HTTPS, lihat [Menandatangani Permintaan API AWS](#) di Referensi Umum AWS.

Informasi lain

Untuk informasi selengkapnya tentang penggunaan AWS STS dengan AWS layanan lain, lihat tautan berikut:

- Amazon S3. Lihat [Membuat permintaan menggunakan kredensi sementara pengguna IAM atau Membuat permintaan menggunakan kredensial sementara pengguna gabungan di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon](#).
- Amazon SNS. Lihat [Mengggunakan kebijakan berbasis identitas dengan Amazon SNS](#) di Panduan Pengembang Layanan Pemberitahuan Sederhana Amazon.
- Amazon SQS. Lihat [Manajemen identitas dan akses di Amazon SQS](#) di Panduan Pengembang Layanan Antrian Sederhana Amazon.
- Amazon SimpleDB. Lihat [Mengggunakan Kredensial Keamanan Sementara](#) di Panduan Pengembang Amazon SimpleDB.

Mengontrol izin untuk kredensial keamanan sementara

Anda dapat menggunakan AWS Security Token Service (AWS STS) untuk membuat dan menyediakan kredensial keamanan sementara yang dapat mengontrol akses ke sumber daya AWS Anda. Untuk informasi lebih lanjut tentang AWS STS, lihat [Kredensial 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-topik berikut ini mengasumsikan bahwa Anda memiliki pengetahuan kerja atas izin dan kebijakan AWS. Untuk informasi lebih lanjut pada topik ini, lihat [Manajemen akses untuk sumber daya AWS](#).

Topik

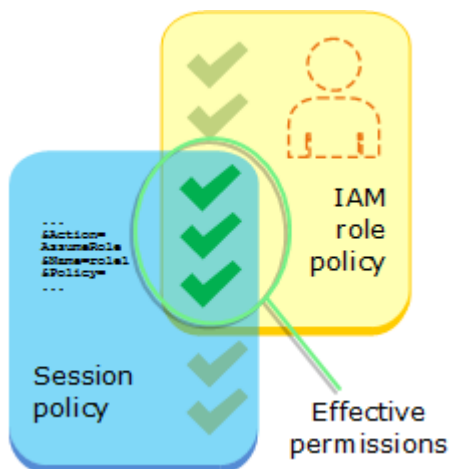
- [Izin untuk AssumeRole, AssumeRoleWith SALL, dan AssumeRoleWithWebIdentity](#)
- [Memantau dan mengontrol tindakan yang diambil dengan peran yang diasumsikan](#)
- [Izin untuk GetFederationToken](#)
- [Izin untuk GetSessionToken](#)
- [Menonaktifkan izin untuk kredensial keamanan sementara](#)

- [Memberikan izin untuk membuat kredensial keamanan sementara](#)

Izin untuk AssumeRole, AssumeRoleWith SALL, 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 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 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 kredensial yang membuat panggilan awal ke AssumeRole tidak dievaluasi oleh AWS saat mengambil keputusan otorisasi "izinkan" atau "tolak". Pengguna untuk sementara menyerahkan izin asli yang mendukung izin yang ditetapkan oleh peran yang diasumsikan. Dalam hal operasi API AssumeRoleWithSAML dan AssumeRoleWithWebIdentity, tidak ada kebijakan untuk dievaluasi karena pemanggil API bukanlah sebuah identitas AWS.

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

```
{
  "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**

```
{
  "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/*"
    }
  ]
}
```

Kebijakan berbasis sumber daya

Beberapa sumber daya AWS mendukung kebijakan berbasis sumber daya, dan kebijakan ini menyediakan mekanisme lain untuk menentukan izin yang memengaruhi kredensial 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

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Principal": {"AWS": "*"},
    "Effect": "Deny",
    "Action": "s3:DeleteObject",
    "Resource": "arn:aws:s3:::productionapp/*"
  }
}
```

Untuk informasi lebih lanjut tentang bagaimana beberapa jenis kebijakan digabungkan dan dievaluasi oleh AWS, lihat [Logika evaluasi kebijakan](#).

Memantau dan mengontrol tindakan yang diambil dengan peran yang diasumsikan

[Peran IAM adalah objek di IAM yang diberi izin](#). Saat Anda [mengambil peran tersebut](#) menggunakan identitas IAM atau identitas dari luar AWS, Anda menerima sesi dengan izin yang ditetapkan ke 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. Saat administrator meninjau aktivitas CloudTrail, mereka dapat melihat informasi

identitas sumber untuk menentukan siapa atau tindakan apa yang dilakukan dengan sesi peran yang diasumsikan.

Setelah identitas sumber ditetapkan, identitas tersebut 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 CLI atau AWS API, yang dikenal sebagai [rantai peran](#). Nilai yang ditetapkan tidak dapat diubah selama sesi peran. Administrator dapat mengonfigurasi izin terperinci 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 pengguna federasi. Untuk mempelajari lebih lanjut tentang tanda sesi, lihat [Melewati tag sesi di AWS STS](#).
- Nama sesi peran — Anda dapat menggunakan kunci `sts:RoleSessionName` kondisi dalam kebijakan kepercayaan peran untuk mengharuskan pengguna Anda 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 penambangan 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 dapat mengakses sumber daya Anda AWS 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. Konfigurasi pengguna dan peran pengujian — Menggunakan lingkungan praproduksi, konfigurasi pengguna dan peran pengujian, serta konfigurasi kebijakan mereka untuk memungkinkan pengaturan identitas sumber.

Jika Anda menggunakan penyedia identitas (IdP) untuk identitas federasi Anda, konfigurasi 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 siapkan untuk pengujian.
3. Tinjauan CloudTrail — Tinjau informasi identitas sumber untuk peran pengujian Anda di CloudTrail log Anda.
4. Latih pengguna Anda — Setelah Anda menguji di lingkungan praproduksi, pastikan pengguna mengetahui cara meneruskan informasi identitas sumber, jika perlu. Tetapkan tenggat waktu kapan Anda akan meminta pengguna Anda untuk memberikan identitas sumber di lingkungan produksi Anda.
5. Konfigurasi kebijakan produksi — Konfigurasi kebijakan Anda untuk lingkungan produksi, lalu tambahkan ke pengguna dan peran produksi Anda.

6. Pantau 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 64 karakter panjang, dapat berisi hanya 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 oleh CloudTrail ketika AWS layanan atau peran terkait layanan melakukan tindakan atas nama identitas federasi atau tenaga kerja.

Important

Anda tidak dapat beralih ke peran AWS Management Console yang memerlukan identitas sumber untuk disetel saat peran diasumsikan. Untuk mengambil peran seperti itu, Anda dapat menggunakan AWS API AWS CLI atau 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 akun yang sama. `Developer_Role` Tetapi Anda ingin mengizinkan tindakan ini hanya jika pengguna telah mengatur identitas sumber ke nama pengguna IAM mereka. Anda dapat melampirkan kebijakan berikut ke pengguna IAM.

Example Contoh kebijakan berbasis identitas yang dilampirkan `DevUser`

```
{
  "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

```

{
  "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 `IAMDevUser`, pengguna mencoba untuk mengasumsikan `DeveloperRole` menggunakan permintaan berikut. AWS CLI

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 dapat 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 operasi tersebut `AssumeRoleWithWebIdentity`. 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 kredensi sementara yang dapat Anda gunakan untuk mengakses AWS sumber daya. Anda dapat menggunakan kredensi pengguna atau peran IAM untuk menelepon. `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 AssumeRoleWith SAFL

Penanggung jawab yang memanggil operasi AssumeRoleWithSAML diautentikasi menggunakan federasi berbasis SAML. Operasi ini mengembalikan satu set kredensi sementara yang dapat Anda gunakan untuk mengakses AWS sumber daya. Untuk informasi selengkapnya tentang penggunaan federasi berbasis SAML untuk AWS Management Console akses, lihat [Mengaktifkan pengguna gabungan SAML 2.0 untuk mengakses AWS Management Console](#) Untuk detail tentang AWS CLI atau akses AWS API, lihat [Federasi SAML 2.0](#). Untuk tutorial menyiapkan federasi SAFL untuk pengguna Active Directory Anda, lihat [AWS Otentikasi 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 SAFL di IAM.](#)
3. [Konfigurasi peran dan izinnya AWS untuk pengguna federasi 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

Pemanggilan utama AssumeRoleWithWebIdentity operasi 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 penggunaan federasi OIDC untuk AWS Management Console akses, lihat.

[Federasi 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": "johndoe",
  "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:](#) ada dalam permintaan. Setelah identitas sumber disetel, 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 Anda ingin meminta pengembang Anda untuk menetapkan identitas sumber untuk mengambil peran penting yang memiliki izin untuk menulis ke AWS sumber daya penting produksi. Juga bayangkan 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 (SALL)

```
{
  "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://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 syarat untuk `sts:SourceIdentity` itu membutuhkan identitas sumber Saanvi atau Diego untuk mengambil peran penting.

Atau, jika Anda menggunakan penyedia OIDC untuk federasi dan pengguna diautentikasi `AssumeRoleWithWebIdentity`, kebijakan kepercayaan peran Anda mungkin terlihat sebagai berikut.

Example Contoh kebijakan kepercayaan peran untuk identitas sumber (penyedia OIDC)

```

{
  "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`

```
{
  "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`

```
{
  "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 kredensial sesi peran yang diperoleh dari asumsi. CriticalRole Identitas sumber ditetapkan selama asumsi CriticalRole, sehingga tidak perlu diatur 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 pengguna federasi. Anda juga dapat melihat peran atau permintaan pengguna untuk mengambil tindakan dalam AWS. File CloudTrail log mencakup informasi tentang identitas sumber yang ditetapkan untuk peran yang diasumsikan atau sesi pengguna federasi. Lihat informasi yang lebih lengkap di [Mencatat panggilan IAM dan AWS STS 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 log AWS CloudTrail

```
"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 boto-core/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 ada di `userIdentity` kunci di CloudTrail log.

Example Contoh kunci UserIdentity dalam log AWS CloudTrail

```
{
  "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, lihat [Contoh peristiwa API IAM dalam catatan CloudTrail](#). Untuk detail selengkapnya tentang informasi yang terkandung dalam file CloudTrail log, lihat [Referensi CloudTrail Acara](#) di Panduan AWS CloudTrail Pengguna.

Izin untuk GetFederationToken

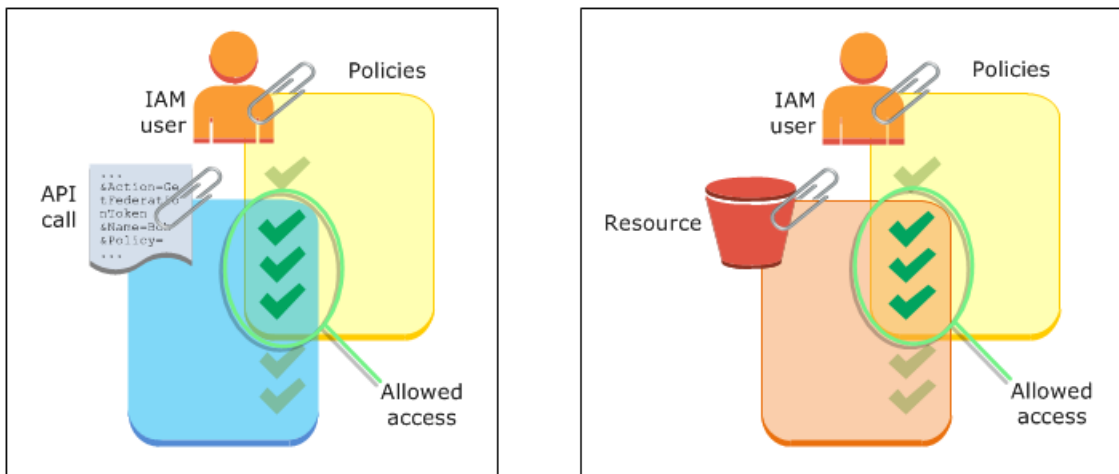
Operasi `GetFederationToken` dipanggil oleh pengguna IAM dan mengembalikan kredensial sementara untuk pengguna tersebut. Operasi ini menggabungkan pengguna. Izin yang diberikan pengguna gabungan ditentukan di salah satu dari dua tempat:

- Kebijakan sesi lewat sebagai sebuah parameter dari Panggilan API `GetFederationToken`. (Ini paling umum.)
- Kebijakan berbasis sumber daya yang secara jelas menyebutkan pengguna gabungan dalam elemen `Principal` dari kebijakan tersebut. (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 gabungan dan melalui kebijakan sesi, izin sesi yang dihasilkan adalah pertemuan dari kebijakan berbasis 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 secara langsung ke layanan AWS dengan kredensial keamanan sementara. Arsitektur ini memungkinkan aplikasi klien untuk membuat permintaan AWS tanpa menyimpan kredensial AWS jangka panjang.

Server autentikasi Anda memanggil `GetFederationToken` API dengan kredensial keamanan jangka panjang dari pengguna IAM yang bernama `token-app`. Tapi jangka panjang kredensial pengguna IAM jangka panjang tetap pada server Anda dan tidak pernah didistribusikan ke klien. Contoh kebijakan berikut terlampir ke pengguna IAM `token-app` dan mendefinisikan kumpulan izin terluas yang akan dibutuhkan oleh pengguna (klien) gabungan Anda. Perhatikan bahwa izin `sts:GetFederationToken` diperlukan bagi layanan autentikasi Anda untuk memperoleh kredensial keamanan sementara bagi pengguna yang digabung.

Note

AWS menyediakan contoh aplikasi Java untuk memenuhi tujuan ini, yang dapat Anda unduh di sini: [Token Vending Machine untuk Pendaftaran Identitas - Sampel Aplikasi Web Java](#).

Example Contoh kebijakan terlampir ke pengguna IAM **token-app** yang memanggil **GetFederationToken**

```
{
  "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 gabungan. Jika pengguna IAM ini memanggil

`GetFederationToken` dan tidak melewati kebijakan sebagai parameter panggilan API, pengguna gabungan yang dihasilkan tidak memiliki izin yang efektif.

Kebijakan sesi diberikan sebagai parameter

Cara paling umum untuk memastikan bahwa pengguna gabungan diberikan izin yang sesuai adalah dengan melewati kebijakan sesi dalam Panggilan API `GetFederationToken`. Memperluas contoh sebelumnya, bayangkan `GetFederationToken` itu dipanggil dengan kredensial dari pengguna `IAMtoken-app`. Bayangkan sesi kebijakan berikut ini diberikan sebagai parameter panggilan API. Pengguna gabungan yang dihasilkan memiliki izin untuk mencantumkan isi bucket Amazon S3 yang bernama `productionapp`. Pengguna tidak dapat melakukan `Amazon S3GetObject`, `PutObject`, dan `DeleteObject` tindakan pada item dalam `productionapp` bucket.

Pengguna gabungan diberi izin ini karena izin tersebut merupakan pertemuan antara kebijakan pengguna IAM dan kebijakan sesi yang Anda lewati.

Pengguna gabungan tidak dapat melakukan tindakan di Amazon SNS Amazon SQS DynamoDB, atau dalam bucket S3 kecuali `productionapp`. Tindakan ini ditolak meskipun izin tersebut diberikan kepada pengguna IAM yang terkait dengan `GetFederationToken` panggilan tersebut.

Example Contoh kebijakan sesi lewat sebagai sebuah parameter dari Panggilan API **`GetFederationToken`**.

```
{
  "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 berbasis sumber daya

Beberapa sumber daya AWS mendukung kebijakan berbasis sumber daya, dan kebijakan ini menyediakan mekanisme lain untuk memberikan izin secara langsung kepada pengguna gabungan. Hanya beberapa layanan AWS 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 “Kebijakan berbasis Sumber Daya” pada tabel. Anda dapat menggunakan kebijakan berbasis sumber daya untuk menetapkan izin secara langsung kepada pengguna gabungan. Lakukan ini dengan menentukan Amazon Resource Name (ARN) dari pengguna gabungan di elemen `Principal` dari 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 mengizinkan pengguna gabungan bernama Carol untuk mengakses bucket. Ketika contoh kebijakan yang dijelaskan sebelumnya dilampirkan pada `token-app` IAM, pengguna gabungan bernama Carol memiliki izin untuk melakukan tindakan `s3:GetObjects`, `s3:PutObject`, dan `s3:DeleteObject` tindakan pada bucket yang bernama `productionapp`. Hal ini berlaku bahkan ketika kebijakan sesi tidak diberikan sebagai parameter panggilan API `GetFederationToken`. Itu karena dalam kasus ini pengguna gabungan bernama Carol telah diberikan izin secara jelas oleh kebijakan berbasis sumber daya berikut.

Ingat, pengguna gabungan diberikan izin hanya ketika izin tersebut secara jelas diberikan kepada pengguna IAM dan pengguna gabungan. Hal tersebut juga bisa diberikan (dalam akun) oleh kebijakan berbasis sumber daya yang secara jelas menyebutkan pengguna gabungan dalam `Principal` elemen kebijakan tersebut, seperti pada contoh berikut.

Example Contoh kebijakan bucket yang mengizinkan akses ke pengguna gabungan

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Principal": {"AWS": "arn:aws:sts::account-id:federated-user/Carol"},
    "Effect": "Allow",
    "Action": [
      "s3:GetObject",
```

```
    "s3:PutObject",
    "s3:DeleteObject"
  ],
  "Resource": ["arn:aws:s3:::productionapp/*"]
}
```

Untuk informasi lebih lanjut 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 AWS Management Console. Untuk melakukan operasi ini, Anda harus memiliki kebijakan yang mengizinkan Anda untuk mengakses tindakan `CreateUser`.

```
{
  "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 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 kredenral, kredenral 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 AWS Management Console dengan meneruskan kredensialnya ke titik akhir masuk tunggal federasi di `https://signin.aws.amazon.com/federation` Untuk informasi selengkapnya, lihat [Mengaktifkan akses broker identitas khusus ke konsol AWS](#).
- 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 [Membandingkan operasi AWS STS API](#)

Untuk informasi selengkapnya tentang akses API yang dilindungi MFA menggunakan `GetSessionToken`, lihat [Mengonfigurasi akses API yang dilindungi MFA](#).

Menonaktifkan izin untuk kredensial keamanan sementara

Kredensial keamanan sementara berlaku hingga kedaluwarsa. Kredensial 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 kredensial ini, tetapi Anda juga harus mengubah izin untuk peran tersebut untuk menghentikan penggunaan kredensial yang disusupi untuk aktivitas akun berbahaya. Izin yang ditetapkan untuk kredensial keamanan sementara dievaluasi setiap kali digunakan untuk membuat permintaan. AWS Setelah Anda menghapus semua izin dari kredensial, AWS permintaan yang menggunakannya gagal.

Mungkin perlu beberapa menit agar pembaruan kebijakan diterapkan. [Cabut kredensial keamanan sementara peran untuk memaksa semua pengguna yang mengasumsikan peran tersebut untuk mengautentikasi](#) ulang dan meminta kredensial baru.

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.

Important

Untuk pengguna di Pusat Identitas IAM, lihat [Menonaktifkan akses pengguna](#) di Panduan AWS IAM Identity Center Pengguna. Anda juga dapat [Hapus akses pengguna](#) ke aplikasi cloud atau aplikasi SAM 2.0 khusus di konsol Pusat Identitas IAM.

Topik

- [Tolak akses ke semua sesi yang terkait dengan peran](#)
- [Tolak akses ke sesi tertentu](#)
- [Tolak sesi pengguna dengan tombol konteks kondisi](#)
- [Tolak pengguna sesi dengan kebijakan berbasis sumber daya](#)

Tolak akses ke semua sesi 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

Prosedur ini menolak izin untuk semua pengguna yang memiliki izin untuk mengambil peran.

Untuk mengubah atau menghapus izin yang ditetapkan ke kredensial keamanan sementara yang diperoleh dengan memanggil `AssumeRole`, `AssumeRoleWithSAML`, atau `AssumeRoleWithWebIdentityGetFederationToken`, atau `GetSessionToken`, Anda dapat mengedit atau menghapus kebijakan izin 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 pengguna sesi dengan kebijakan berbasis sumber daya](#) untuk detail.

1. Masuk ke AWS Management Console dan buka konsol IAM.
2. Di panel navigasi, pilih nama peran yang akan diedit. Anda dapat menggunakan kotak pencarian untuk memfilter daftar.
3. Pilih kebijakan yang relevan.
4. Pilih tab Izin.
5. Pilih tab JSON dan perbarui kebijakan untuk menolak semua sumber daya dan tindakan.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*"
    }
  ]
}
```


6. Di halaman Tinjauan, tinjau Ringkasan kebijakan lalu pilih Simpan perubahan untuk menyimpan pekerjaan Anda.

Saat memperbarui kebijakan, perubahan akan memengaruhi izin semua kredensial keamanan sementara yang terkait dengan peran tersebut, termasuk kredensial yang dikeluarkan sebelum Anda mengubah kebijakan izin peran. Setelah memperbarui kebijakan, Anda dapat [mencabut kredensial keamanan sementara peran tersebut untuk segera mencabut semua izin atas kredensial](#) yang dikeluarkan peran tersebut.

Tolak akses ke sesi tertentu

Saat Anda memperbarui peran yang dapat diasumsikan dari iDP dengan kebijakan penolakan semua atau menghapus peran sepenuhnya, semua pengguna yang memiliki akses ke peran tersebut akan terganggu. Anda dapat menolak akses berdasarkan Principal elemen 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 ARN pengguna federasi menggunakan AWS CloudTrail log. Untuk informasi selengkapnya, lihat [Cara Mudah Mengidentifikasi Pengguna Federasi Anda dengan Menggunakan AWS CloudTrail](#).

Tolak sesi pengguna dengan tombol konteks kondisi

Anda dapat menggunakan kunci konteks kondisi dalam situasi di mana Anda ingin menolak akses ke sesi kredensial keamanan sementara tertentu tanpa memengaruhi izin pengguna IAM atau peran yang membuat kredensialnya.

Untuk informasi selengkapnya tentang kunci konteks kondisi, lihat [AWS kunci konteks kondisi global](#).

 Note

Jika ada kebijakan berbasis sumber daya yang memungkinkan akses utama, Anda juga harus menambahkan pernyataan penolakan eksplisit pada kebijakan berbasis sumber daya setelah Anda menyelesaikan langkah-langkah ini.

Setelah memperbarui kebijakan, Anda dapat [mencabut kredensial keamanan sementara peran tersebut untuk segera mencabut semua kredensial](#) yang dikeluarkan.

aws: PrincipalArn

Anda dapat menggunakan kunci konteks kondisi [aws: PrincipalArn](#) untuk menolak akses ke ARN utama tertentu. Anda melakukannya dengan menentukan pengenal unik (ID) pengguna IAM, peran, atau pengguna federasi yang terkait dengan kredensial keamanan sementara dalam elemen Kondisi kebijakan.

1. Di panel navigasi konsol IAM, pilih nama peran yang akan diedit. Anda dapat menggunakan kotak pencarian untuk memfilter daftar.
2. Pilih kebijakan yang relevan.
3. Pilih tab Izin.
4. Pilih tab JSON dan tambahkan pernyataan penolakan untuk ARN utama seperti yang ditunjukkan pada contoh berikut.

```
{
  "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:sts::222222222222:federated-user/USERNAME"
          ]
        }
      }
    }
  ]
}
```

5. Di halaman Tinjauan, tinjau Ringkasan kebijakan lalu pilih Simpan perubahan untuk menyimpan pekerjaan Anda.

aws:userid

Anda dapat menggunakan kunci konteks kondisi [aws:userid](#) 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 pengguna federasi yang terkait dengan kredensial 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 pengenal unik untuk pengguna IAM. Menentukan pengenal unik pengguna IAM sebagai nilai untuk kunci konteks `aws:userid` akan menolak semua sesi yang terkait dengan pengguna IAM.
- `AROAXROLE1` mewakili pengidentifikasi unik untuk peran IAM. Menentukan pengenal unik peran IAM sebagai nilai untuk kunci konteks `aws:userid` akan menolak semua sesi yang terkait dengan peran tersebut.
- `AROAXROLE2` mewakili pengenal unik untuk sesi peran yang diasumsikan. Di bagian `caller-specified-role-session -name` dari pengidentifikasi 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 kredensial. Jika Anda menentukan wildcard untuk nama sesi peran, itu akan menolak semua sesi yang terkait dengan peran tersebut.
- `account-id:<federated-user-caller-specified-name>` mewakili pengenal unik untuk sesi pengguna federasi. Pengguna federasi dibuat oleh pengguna IAM yang memanggil `API. GetFederationToken` Jika Anda menentukan pengenal unik untuk pengguna federasi, itu akan menolak sesi pengguna federasi bernama tanpa memengaruhi izin peran yang membuat kredensialnya.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "aws:userId": [
```

```

        "AIDAXUSER1",
        "AROAXROLE1",
        "AROAXROLE2:<caller-specified-role-session-name>",
        "account-id:<federated-user-caller-specified-name>"
    ]
}
}
}
]
}

```

Untuk contoh spesifik dari nilai kunci utama, lihat [Nilai-nilai kunci utama](#). Untuk informasi tentang pengidentifikasi unik IAM, lihat [Pengidentifikasi unik](#)

Tolak pengguna sesi dengan kebijakan berbasis sumber daya

Jika ARN utama juga termasuk dalam kebijakan berbasis sumber daya, Anda juga harus mencabut akses berdasarkan nilai pengguna `principalId` atau `sourceIdentity` nilai tertentu dalam elemen kebijakan berbasis sumber daya. `Principal` Jika Anda hanya memperbarui kebijakan izin untuk peran tersebut, pengguna tetap dapat melakukan tindakan yang diizinkan 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 AWS Management Console dan buka konsol untuk layanan ini. Setiap layanan memiliki lokasi yang berbeda di konsol untuk melampirkan kebijakan.
3. Edit pernyataan kebijakan untuk menentukan informasi identifikasi kredensi:
 - a. Di `Principal`, masukkan ARN kredensi untuk ditolak.
 - b. Masuk `Effect`, masukkan "Deny."
 - c. Masuk `Action`, masukkan namespace layanan dan nama tindakan yang akan ditolak. Untuk menolak semua tindakan, gunakan karakter wildcard (*). Misalnya: "s3: *."
 - d. Masuk `Resource`, masukkan ARN dari sumber daya target. Misalnya: "arn:aws:s3: ::EXAMPLE-BUCKET."

```

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

```

```
"Principal": [  
    "arn:aws:iam::222222222222:role/ROLENAME",  
    "arn:aws:iam::222222222222:user/USERNAME",  
    "arn:aws:sts::222222222222:federated-user/USERNAME"  
],  
"Effect": "Deny",  
"Action": "s3:*",  
"Resource": "arn:aws:s3:::EXAMPLE-BUCKET"  
}  
}
```

4. Simpan pekerjaan Anda.

Memberikan izin untuk membuat kredensial keamanan sementara

Secara default, pengguna IAM tidak memiliki izin untuk membuat kredensial keamanan sementara untuk pengguna gabungan dan peran. 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 kredensial keamanan sementara bagi pengguna gabungan atau peran, Anda melampirkan kebijakan yang memberikan salah satu atau kedua hak istimewa berikut:

- Bagi pengguna gabungan untuk mengakses peran IAM, berikan akses ke `AWS STSAssumeRole`.
- Bagi pengguna gabungan yang tidak membutuhkan peran, berikan akses ke `AWS STS GetFederationToken`.

Untuk informasi selengkapnya tentang perbedaan antara operasi API `AssumeRole` dan `GetFederationToken`, lihat [Meminta kredensial 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

Kebijakan contoh berikut memberikan izin `AssumeRole` untuk memanggil `UpdateApp` peran dalam Akun `AWS123123123123`. 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.

```
{
  "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`.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": "sts:GetFederationToken",
    "Resource": "*"
  }]
}
```

Important

Saat Anda memberikan pengguna IAM izin untuk membuat kredensial keamanan sementara bagi pengguna gabungan dengan `GetFederationToken`, ketahuilah bahwa ini mengizinkan pengguna tersebut untuk mengeluarkan izin milik mereka. Untuk informasi selengkapnya tentang pendelegasian izin terhadap seluruh pengguna IAM dan Akun AWS, lihat [Contoh kebijakan untuk mendelegasikan akses](#). Untuk informasi selengkapnya tentang mengatur izin pada kredensial keamanan sementara, lihat [Mengontrol 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 membiarkan pengguna IAM membuat kredensial keamanan sementara hanya untuk pengguna gabungan yang namanya dimulai dengan Manajer.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": "sts:GetFederationToken",
    "Resource": ["arn:aws:sts::123456789012:federated-user/Manager*"]
  }]
}
```

Mengelola AWS STS dalam Wilayah AWS

Secara default, AWS Security Token Service (AWS STS) tersedia sebagai layanan global, dan semua AWS STS permintaan masuk ke satu titik akhir di `https://sts.amazonaws.com`. AWS merekomendasikan penggunaan AWS STS titik akhir Regional alih-alih titik akhir global untuk mengurangi latensi, membangun redundansi, dan meningkatkan validitas token sesi.

- Mengurangi 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 endpoint 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 AWS Well-Architected Framework.
- 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 yang diaktifkan secara default. Jika Anda bermaksud 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.

Mengelola token sesi titik akhir global

Sebagian Wilayah AWS besar diaktifkan untuk operasi di semua secara Layanan AWS default. Wilayah tersebut secara otomatis diaktifkan untuk digunakan dengan AWS STS. Beberapa Wilayah, seperti Asia Pacific (Hong Kong) harus diaktifkan secara manual. Untuk mempelajari lebih lanjut tentang mengaktifkan dan menonaktifkan Wilayah AWS, lihat [Menentukan Wilayah AWS akun yang dapat digunakan](#) dalam Panduan Referensi AWS Account Management . Ketika Anda mengaktifkan AWS Wilayah ini, mereka secara otomatis diaktifkan untuk digunakan dengan AWS STS. Anda tidak dapat mengaktifkan AWS STS titik akhir untuk Wilayah yang dinonaktifkan. Token yang valid di semua Wilayah AWS menyertakan 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 AWS Management Console, 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 Security Token Service (STS) bagian Token Sesi dari titik akhir STS. Titik akhir Global menunjukkan `Valid only in Wilayah AWS enabled by default`. Pilih Ubah.
4. Dalam kotak dialog Ubah kompatibilitas wilayah, pilih Semua Wilayah AWS. Lalu pilih Simpan Perubahan.

Note

Token yang valid di semua Wilayah AWS menyertakan 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 kesesuaian Wilayah token sesi untuk titik akhir global (AWS CLI)

Atur versi token sesi. Token versi 1 hanya valid dalam 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 kesesuaian Wilayah token sesi untuk titik akhir global (API AWS)

Atur versi token sesi. Token versi 1 hanya valid dalam 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](#)

Mengaktifkan dan menonaktifkan AWS STS dalam Wilayah AWS

Saat Anda mengaktifkan titik akhir STS untuk Wilayah, AWS STS dapat mengeluarkan kredensial sementara untuk pengguna dan peran di akun Anda yang membuat permintaan. AWS STS 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 titik akhir STS Regional di akun tempat kredensial sementara dihasilkan. Tidak masalah apakah pengguna masuk ke akun yang sama atau akun yang berbeda saat mereka mengajukan permintaan. Untuk Wilayah yang diaktifkan secara manual, Anda harus mengaktifkan Wilayah di akun yang membuat permintaan dan akun tempat kredensial sementara dibuat.

Misalnya, bayangkan pengguna di akun A ingin mengirim permintaan `sts:AssumeRole` API ke titik akhir AWS STS `https://sts.us-east-2.amazonaws.com` Regional. Permintaan adalah untuk kredensial sementara untuk peran bernama `Developer` di akun B. karena permintaan adalah untuk membuat kredensial untuk entitas di akun B, akun B harus mengaktifkan Wilayah `us-east-2`. Pengguna dari akun A (atau akun lain) dapat memanggil titik akhir `us-east-2` untuk meminta kredensial bagi akun B apakah Wilayah diaktifkan di akun mereka atau tidak.

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 Security Token Service (STS) Endpoints, cari Region yang ingin Anda konfigurasi, lalu pilih Active or Inactive di kolom status STS.
4. Di kotak dialog yang terbuka, pilih Aktifkan atau Nonaktifkan.

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 menggunakan AWS STS Wilayah

Setelah mengaktifkan Region, Anda dapat mengarahkan panggilan AWS STS API ke Region tersebut. Cuplikan kode Java berikut menunjukkan cara mengonfigurasi objek `AWSecurityTokenService` untuk membuat permintaan ke Wilayah Eropa (Irlandia) (`eu-west-1`).

```
EndpointConfiguration regionEndpointConfig = new EndpointConfiguration("https://sts.eu-west-1.amazonaws.com", "eu-west-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 [Menentukan Wilayah AWS akun yang dapat digunakan](#) dalam Panduan AWS Account Management Referensi.









Dalam contoh, baris pertama membuat instance `EndpointConfiguration` objek yang dipanggil `regionEndpointConfig`, melewati 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](#).

















Wilayah dan titik akhir









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

Nama wilayah	Titik akhir	Aktif secara default	Mengaktifkan/menonaktifkan secara manual
AS Barat (Oregon)	sts.us-west-2.amazonaws.com	 Ya	 Ya
Afrika (Cape Town)	sts.af-south-1.amazonaws.com	 Tidak	 Tidak
Asia Pasifik (Hong Kong)	sts.ap-east-1.amazonaws.com	 Tidak	 Tidak
Asia Pasifik (Hyderabad)	sts.ap-south-2.amazonaws.com	 Tidak	 Tidak
Asia Pasifik (Jakarta)	sts.ap-southeast-3.amazonaws.com	 Tidak	 Tidak
Asia Pasifik (Melbourne)	sts.ap-southeast-4.amazonaws.com	 Tidak	 Tidak
Asia Pasifik (Mumbai)	sts.ap-south-1.amazonaws.com	 Ya	 Ya
Asia Pasifik (Osaka)	sts.ap-northeast-3.amazonaws.com	 Ya	 Ya

Nama wilayah	Titik akhir	Aktif secara default	Mengaktifkan/menonaktifkan secara manual
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 (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	 Ya	 Ya
Tiongkok (Beijing)	sts.cn-north-1.amazonaws.com.cn	 Ya	 Tidak
Tiongkok (Ningxia)	sts.cn-northwest-1.amazonaws.com.cn	 Ya	 Ya

Nama wilayah	Titik akhir	Aktif secara default	Mengaktifkan/menonaktifkan secara manual
Eropa (Frankfurt)	sts.eu-central-1.amazonaws.com	 Ya	 Ya
Eropa (Irlandia)	sts.eu-west-1.amazonaws.com	 Ya	 Ya
Eropa (London)	sts.eu-west-2.amazonaws.com	 Ya	 Ya
Eropa (Milan)	sts.eu-south-1.amazonaws.com	 Tidak	 Tidak
Eropa (Paris)	sts.eu-west-3.amazonaws.com	 Ya	 Ya
Eropa (Spanyol)	sts.eu-south-2.amazonaws.com	 Tidak	 Tidak
Eropa (Stockholm)	sts.eu-north-1.amazonaws.com	 Ya	 Ya
Eropa (Zürich)	sts.eu-central-2.amazonaws.com	 Tidak	 Tidak

Nama wilayah	Titik akhir	Aktif secara default	Mengaktifkan/menonaktifkan secara manual
Israel (Tel Aviv)	sts.il-central-1.amazonaws.com	 Tidak	 Tidak
Timur Tengah (Bahrain)	sts.me-south-1.amazonaws.com	 Tidak	 Tidak
Timur Tengah (UEA)	sts.me-central-1.amazonaws.com	 Tidak	 Tidak
Amerika Selatan (Sao Paulo)	sts.sa-east-1.amazonaws.com	 Ya	 Ya

¹Anda harus [mengaktifkan Wilayah](#) untuk menggunakannya. Ini secara otomatis mengaktifkan. AWS STS Anda tidak dapat mengaktifkan atau menonaktifkan secara manual AWS STS di Wilayah ini.

²Untuk digunakan AWS di China, Anda memerlukan akun dan kredensial khusus di China. AWS

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

Untuk informasi selengkapnya, lihat [Mencatat panggilan IAM dan AWS STS API dengan AWS CloudTrail](#).

Menggunakan token pembawa

Beberapa layanan AWS mengharuskan Anda memiliki izin untuk mendapatkan token pembawa layanan AWS STS sebelum Anda dapat mengakses sumber daya mereka secara terprogram. Layanan ini mendukung protokol yang mengharuskan Anda menggunakan token pembawa sebagai ganti dari penggunaan [Permintaan tanda tangan Signature Versi 4](#) biasa. Saat Anda melakukan operasi API AWS CLI atau AWS yang memerlukan token pembawa, layanan AWS meminta token pembawa atas nama Anda. Layanan memberi Anda token, yang kemudian dapat Anda gunakan untuk melakukan operasi selanjutnya dalam layanan tersebut.

Token pembawa layanan AWS STS mencakup informasi dari autentikasi utama asli yang mungkin 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 `aws codeartifact get-authorization-token` operasi. 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:

```
{  
  "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](#) di panduan AWS CodeArtifact pengguna.

Contoh aplikasi yang menggunakan kredensial sementara

Anda dapat menggunakan AWS Security Token Service (AWS STS) untuk membuat dan menyediakan kredensial keamanan sementara kepada pengguna tepercaya yang dapat mengontrol akses ke sumber daya Anda AWS. Untuk informasi selengkapnya tentang AWS STS, lihat [Kredensial 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 SAFL 2.0](#). Menunjukkan cara menghapus akses menggunakan federasi perusahaan untuk AWS menggunakan Windows Active Directory (AD), Active Directory Federation Services (ADFS) 2.0, dan SAFL (Security Assertion Markup Language) 2.0.
- [Mengaktifkan akses broker identitas khusus ke konsol AWS](#). Menunjukkan cara membuat proxy federasi kustom yang memungkinkan single sign-on (SSO) sehingga pengguna Active Directory yang ada dapat masuk ke file. AWS Management Console
- [Cara Menggunakan Shibboleth untuk Single Sign-On ke AWS Management Console](#). Menunjukkan cara menggunakan [Shibboleth](#) dan [SAML](#) untuk memberikan akses pengguna dengan sistem masuk tunggal (SSO) ke AWS Management Console.

Sampel untuk federasi OIDC

Contoh aplikasi berikut menggambarkan cara menggunakan OIDC Federation dengan penyedia seperti Login with Amazon, Amazon Cognito, Facebook, atau Google. Anda dapat menukar otentikasi dari penyedia ini untuk kredensial AWS keamanan sementara untuk mengakses layanan. AWS

- [Tutorial Amazon Cognito](#) — Kami menyarankan Anda menggunakan Amazon Cognito dengan SDK AWS 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 [Otentikasi dengan Amplify di Dokumentasi Amplify](#).

Mengaktifkan akses broker identitas khusus ke konsol AWS

Anda dapat menulis dan menjalankan kode untuk membuat URL yang memungkinkan pengguna yang masuk ke jaringan organisasi Anda mengakses AWS Management Console. URL menyertakan token masuk yang Anda dapatkan AWS dan yang mengautentikasi pengguna. AWS Sesi konsol yang dihasilkan mungkin mencakup perbedaan AccessKeyId karena federasi. Untuk melacak penggunaan kunci akses untuk login federasi melalui CloudTrail acara terkait, lihat [Mencatat panggilan IAM dan AWS STS API dengan AWS CloudTrail](#) dan acara [AWS Management Console login](#).

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 pengguna gabungan SAML 2.0 untuk mengakses AWS Management Console](#).

Untuk memungkinkan pengguna organisasi Anda mengakses AWS Management Console, Anda dapat membuat pialang 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 [Menggunakan peran IAM](#). Untuk mempelajari cara mengirimkan tag sesi opsional saat Anda mendapatkan kredensial keamanan Anda, lihat [Melewati 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. Ketika Anda menggunakan `DurationSeconds` dalam `AssumeRole*` operasi, Anda harus menyebutnya sebagai pengguna IAM dengan kredensi jangka panjang. Jika tidak, panggilan ke titik akhir federasi pada langkah 3 gagal. Untuk mempelajari cara melihat atau mengubah nilai maksimum untuk peran, lihat [Lihat pengaturan 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. Hubungi titik akhir AWS federasi dan berikan kredensi 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

- Jangan gunakan parameter `HTTP SessionDuration` jika Anda mendapatkan kredensial sementara dengan `GetFederationToken`. Melakukan hal itu akan menyebabkan kegagalan operasi.
- Menggunakan kredensi 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.

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 AWS Management Console jika Anda telah mengaktifkan izin di kredensial 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 lebih lanjut tentang kode status respons HTTP 302, kunjungi [RFC 2616, bagian 10.3.3](#).

Untuk menyelesaikan tugas-tugas ini, Anda dapat menggunakan [HTTPS Query API for 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-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 memberi pengguna federasi Anda akses langsung ke file. AWS Management Console 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 akses pengguna federasi ke sumber daya Anda dari AWS Management Console

1. Autentikasi pengguna di sistem identitas dan otorisasi Anda.
2. Dapatkan kredensial keamanan sementara untuk pengguna. Kredensi 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 kredensi sementara, Anda memanggil AWS STS [AssumeRole](#) API (disarankan) atau API. [GetFederationToken](#) Untuk informasi selengkapnya tentang perbedaan antara operasi API ini, lihat [Memahami Opsi API untuk Mendelegasikan Akses ke AWS Akun Anda dengan Aman](#) 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 [Mengontrol 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 mengkode 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 kemungkinan nilai *kode wilayah*, lihat kolom Wilayah di titik akhir [AWS Masuk](#). Anda dapat secara opsional menggunakan titik akhir federasi AWS Masuk 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 tidak ada, maka sesi default ke durasi kredensial yang Anda ambil dari AWS STS langkah 2 (yang defaultnya 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

- Jangan gunakan parameter `HTTP SessionDuration` jika Anda mendapatkan kredensial sementara dengan `GetFederationToken`. Melakukan hal itu akan menyebabkan kegagalan operasi.
- Menggunakan kredensi 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.

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 kredensial 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
Aj0srkdPkghomIe3mJip5%2F0djDBbo7Sm0%2FENDEiCdpsQKodTpleKA8xQq0CwFg6a69xdEBQT8
FipATnLbKoyS4b%2FebhnsTUjZZQWp0wXXqFF7gSm%2FMe2tXe0jzsdP0012obez9lijPSdF1k2b5
PFGhiuyAR9aD5%2BubM0pY86fKex1qsyTjvyTbZ9nXe6DvxVDcnC0h0GETJ7XfKSFdH0v%2FYR25C
UAhJ3nXIkIbG7Ucv9c0EpCf%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 oleh pengguna gabungan Anda untuk mengakses AWS Management Console. 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_w2BE5HYexbe9y40f-kje53SsjNNecATfjIzpw1  
WibbnH6YcYRiBoffZBGExbEXAMPLE5aiKX4THWjQKC6gg6alHu6JFrn0JoK3dtP6I9a6hi6yPgm  
i0kPZMmNGmhsVxetKzr8mx3pxhHbMEEXAMPLETv1pij0rok3IyCR2YVcIjqwfWv32HU2X1j471u  
3fU6u0fUComeKiqTGX974xzJ0ZbdmX_t_1LrhEXAMPLEDDIisSnyHGw2xaZZqudm4mo2uTDk9Pv
```

```
915K0ZCqIgEXAMPLEcA6tgLPykEWGuYH6BdSC6166n4M4JkXIQgac7_7821YqixsNxZ6rsrpzwf
nQoS1407R0eJCCJ684EXAMPLEZRdBNnuLbUYpz2Iw3vIN0tQg0ujwnwydPscM9F7foaEK3jwMkg
Apeb1-6L_0B12MZhuFxx55555EXAMPLEehyETEd4Zu1KPdXHkg16T9Zk11Hz2Uy1RUTUhhUxNtSQ
nWc5xkbBoEcXqpoSIEk7yhje9Vzhd61AEXAMPLE1bWeouACEMG6-Vd3dAgFYd6i5FYoyFrZLWvm
0LSG7RyYKeYN5VIzUk3YWQpyjP0RiT5KURsUi-NEXAMPLExMOMdo0DBEgKQsk-iu2ozh6r8bxwC
RNhujg
```

Contoh kode menggunakan Python

Contoh berikut menunjukkan bagaimana menggunakan Python untuk secara terprogram membangun URL yang memberikan pengguna federasi akses langsung ke file. AWS Management Console Berikut ini adalah dua contoh:

- Federasi melalui GET permintaan ke AWS
- Federasi melalui permintaan POST untuk AWS

Kedua contoh menggunakan [AssumeRole](#) API [AWS SDK for Python \(Boto3\)](#) dan untuk mendapatkan kredensial keamanan sementara.

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 federated user

# 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 AAKun AWS,
# call "AssumeRole" to get temporary access keys for the federated user

# 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);
```

Contoh kode menggunakan Java

Contoh berikut menunjukkan cara menggunakan Java untuk secara terprogram menyusun URL yang memberikan akses langsung kepada pengguna gabungan ke AWS Management Console. Potongan 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 bufferedReader = new BufferedReader(new
    InputStreamReader(conn.getInputStream()));
String returnContent = bufferedReader.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 cara menggunakan Ruby untuk secara terprogram menyusun URL yang memberikan akses langsung kepada pengguna gabungan ke AWS Management Console. 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
```

Sumber daya tambahan untuk kredensial keamanan sementara

Skenario dan aplikasi berikut dapat memandu Anda dalam menggunakan kredensial keamanan sementara:

- [Bagaimana mengintegrasikan 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
- [Mengonfigurasi akses API yang dilindungi 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 sumber daya AWS](#)
- [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?](#).

Menandai sumber daya IAM

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 di sumber daya IAM, lihat [IAM dan AWS STS kuota](#).

Note

Untuk detail tentang sensitivitas huruf besar untuk kunci tag dan nilai kunci tag, lihat [Case sensitivity](#).

Tag membantu Anda mengidentifikasi dan mengatur sumber daya AWS. Banyak layanan AWS yang mendukung penandaan, sehingga Anda dapat menetapkan tag yang sama ke sumber daya dari layanan yang berbeda untuk menunjukkan bahwa sumber daya tersebut 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 [Menandai AWS sumber daya](#) Panduan Pengguna.

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 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 di AWS STS untuk menambahkan atribut khusus saat Anda mengasumsikan peran atau memfederasi pengguna. Untuk informasi selengkapnya, lihat [Melewati tag sesi di AWS STS](#).

Pilih konvensi penamaan tanda AWS

Saat Anda mulai melampirkan tag ke sumber daya IAM Anda, pilih konvensi penamaan tag Anda dengan hati-hati. Terapkan konvensi yang sama ke semua tanda AWS Anda. Hal ini penting jika Anda menggunakan tanda dalam kebijakan untuk mengontrol akses ke sumber daya AWS. 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](#) dalam panduan pengguna Organisasi untuk mempelajari lebih lanjut tentang penggunaan tag di Organisasi.

Praktik terbaik untuk penamaan tag

Ini adalah beberapa praktik terbaik dan konvensi penamaan untuk tag.

Pastikan bahwa nama tag digunakan secara konsisten. Misalnya, tag `CostCenter` dan `costcenter` berbeda, jadi satu dapat dikonfigurasi sebagai tag alokasi biaya untuk analisis keuangan dan pelaporan dan yang lainnya mungkin tidak. Demikian pula, Nama tag muncul di AWS Konsol untuk banyak sumber daya, tetapi nama tag tidak. Untuk detail tentang sensitivitas huruf besar untuk kunci tag dan nilai kunci tag, lihat [Case sensitivity](#).

Sejumlah tag telah ditentukan oleh AWS atau dibuat secara otomatis oleh berbagai AWS layanan. Banyak AWS Nama tag -defined menggunakan semua huruf kecil, dengan tanda hubung yang memisahkan kata dalam nama, dan awalan untuk mengidentifikasi layanan sumber untuk tag tersebut. Misalnya:

- `aws:ec2spot:fleet-request-id` mengidentifikasi Permintaan Instans Spot Amazon EC2 yang meluncurkan instans.
- `aws:cloudformation:stack-name` mengidentifikasi AWS CloudFormation tumpukan yang menciptakan sumber daya.
- `elasticbeanstalk:environment-name` mengidentifikasi aplikasi yang menciptakan sumber daya.

Pertimbangkan untuk memberi nama 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 mendefinisikan 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 untuk

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 pada nama tag. Sebagai contoh, `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, AWS STS sesi `assume-role`, dan AWS STS sesi pengguna federasi:

Persyaratan karakter— Tombol dan nilai tag dapat mencakup kombinasi huruf, angka, spasi, dan `._:/ = + - @` simbol.

Sensitivitas kasus— Sensitivitas kasus untuk kunci tag berbeda tergantung pada jenis sumber daya IAM yang ditandai. Nilai kunci tag untuk pengguna dan peran IAM tidak peka huruf besar/kecil, tetapi kasus 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. Kami menyarankan Anda memutuskan strategi untuk memanfaatkan tag, dan secara konsisten menerapkan strategi itu di semua jenis sumber daya. Untuk mempelajari lebih lanjut tentang praktik terbaik untuk penandaan, lihat [Menandai AWS Sumber Daya](#) di Referensi Umum AWS.

Daftar berikut menunjukkan perbedaan sensitivitas kasus untuk kunci tag yang dilampirkan ke sumber daya IAM.

Nilai kunci tanda tidak peka huruf besar kecil:

- IAM role
- 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:**. Prefiks tanda ini dicadangkan untuk penggunaan internal AWS.
- 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 bukan 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 khusus. 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 sebuah 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 dari kunci tag yang sama. Misalnya, jika Anda memiliki pengguna dengan pasangan nilai kunci tag `costCenter = 1234`, Anda kemudian dapat melampirkan pasangan nilai kunci tag `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 menimpa 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**.

Topik

- [Menandai pengguna IAM](#)
- [Menandai peran IAM](#)
- [Menandai kebijakan yang dikelola pelanggan](#)
- [Menandai penyedia identitas IAM](#)
- [Menandai profil instans untuk peran Amazon EC2](#)
- [Menandai sertifikat server](#)
- [Menandai perangkat MFA virtual](#)
- [Melewati tag sesi di AWS STS](#)

Menandai pengguna IAM

Anda dapat menggunakan pasangan kunci-nilai tanda IAM untuk menambahkan atribut kustom 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 di AWS STS untuk menambahkan atribut khusus saat Anda mengasumsikan peran atau memfederasi pengguna. Untuk informasi selengkapnya, lihat [Melewati tag sesi di AWS STS](#).

Izin yang diperlukan untuk memberi tag ke pengguna IAM.

Anda harus mengonfigurasi izin untuk mengizinkan pengguna IAM untuk memberi tag ke 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 untuk menambahkan, membuat daftar, atau menghapus tag untuk pengguna tertentu

Tambahkan pernyataan berikut ini ke kebijakan izin untuk pengguna IAM yang perlu mengelola tag. Gunakan nomor akun Anda dan ganti *<username>* dengan nama pengguna yang tandanya 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 untuk menambahkan tag ke pengguna tertentu

Tambahkan pernyataan berikut ini ke kebijakan izin untuk pengguna IAM yang perlu menambahkan, tapi 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 tandanya 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 yang AWS dikelola seperti [IAMFullAccess](#) untuk memberikan akses penuh ke IAM.

Mengelola tag pada pengguna IAM (konsol)

Anda dapat mengelola tag untuk pengguna IAM dari AWS Management Console.

Untuk mengelola tanda pada pengguna (konsol)

1. Masuk ke AWS Management Console 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 rangkaian tag yang sudah 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 untuk pengguna IAM.

Untuk membuat daftar tag yang saat ini ada pada pengguna IAM (AWS CLI atau AWS API))

- AWS CLI: [AWS list-user-tags](#)
- AWS API: [ListUserTags](#)

Untuk memberi 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 memberi tanda ke sumber daya untuk layanan AWS lainnya, 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 peran IAM

Anda dapat menggunakan pasangan kunci-nilai tanda IAM untuk menambahkan atribut kustom 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 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 di AWS STS untuk menambahkan atribut khusus saat Anda mengasumsikan peran atau memfederasi pengguna. Untuk informasi selengkapnya, lihat [Melewati tag sesi di AWS STS](#).

Izin yang diperlukan untuk menandai peran IAM.

Anda harus mengonfigurasi izin untuk mengizinkan peran untuk memberi tag ke 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 untuk menambahkan, membuat daftar, atau menghapus tag tag tag untuk tag untuk pengguna tertentu

Tambahkan pernyataan berikut ini ke kebijakan izin untuk peran IAM yang perlu mengelola tanda untuk mengelola tag tag. Gunakan nomor akun Anda dan ganti `<username>` dengan nama pengguna yang tandanya 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 tanda ke pengguna tertentu

Tambahkan pernyataan berikut ini ke kebijakan izin untuk peran IAM yang perlu menambahkan, tapi tidak menghapus, tag untuk pengguna tertentu.

 Note

Tindakan `iam:TagRole` mengharuskan Anda untuk juga menyertakan tindakan `iam:ListRoleTags`.

Untuk menggunakan kebijakan ini, ganti `<username>` dengan nama pengguna yang tandanya 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 untuk menambahkan, membuat daftar, atau menghapus tag tag tag untuk peran tertentu

Tambahkan pernyataan berikut ini ke kebijakan izin untuk peran IAM yang perlu mengelola tanda untuk mengelola tag tag. Ganti `<rolename>` dengan nama peran yang tandanya 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 yang AWS dikelola seperti [IAMFullAccess](#) untuk memberikan akses penuh ke IAM.

Mengelola tag pada peran IAM (konsol)

Anda dapat mengelola tanda untuk peran IAM dari AWS Management Console.

Untuk mengelola tanda pada peran (konsol)

1. Masuk ke AWS Management Console 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 belum memiliki tanda.
 - Pilih Kelola tanda untuk mengelola rangkaian tag yang sudah ada.
4. Tambahkan atau hapus tag untuk menyelesaikan rangkaian tag. Kemudian, pilih Simpan perubahan.

Mengelola tag pada peran IAM (AWS CLI atau AWS API)

Anda dapat membuat daftar, atau menghapus tanda untuk peran IAM. Anda dapat menggunakan AWS CLI atau AWS API untuk mengelola tanda untuk peran IAM.

Untuk membuat daftar tag yang saat ini ada pada peran (AWS CLI atau AWS API)

- AWS CLI: [AWS list-role-tags](#)
- AWS API: [ListRoleTags](#)

Untuk memberi 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 memberi tanda ke sumber daya untuk layanan AWS lainnya, 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 kebijakan yang dikelola pelanggan

Anda dapat menggunakan pasangan nilai kunci tanda IAM untuk menambahkan atribut kustom ke kebijakan terkelola pelanggan. 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 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 di AWS STS untuk menambahkan atribut khusus saat Anda mengasumsikan peran atau memfederasi pengguna. Untuk informasi selengkapnya, lihat [Melewati tag sesi di AWS STS](#).

Izin yang diperlukan untuk menandai kebijakan terkelola pelanggan

Anda harus mengonfigurasi izin untuk mengizinkan entitas IAM (pengguna atau peran) untuk menandai kebijakan terkelola 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) untuk menambahkan, membuat daftar, atau menghapus tanda bagi kebijakan terkelola 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 tandanya 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) untuk menambahkan tanda ke kebijakan terkelola pelanggan

Tambahkan pernyataan berikut ini ke kebijakan izin untuk entitas IAM yang perlu menambahkan, tapi 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 tandanya 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 yang AWS dikelola oleh [IAMFullAccess](#) untuk memberikan akses penuh ke IAM.

Mengelola tanda pada kebijakan terkelola pelanggan (konsol)

Anda dapat mengelola tanda untuk kebijakan terkelola pelanggan AWS Management Console.

Untuk mengelola tanda pada kebijakan terkelola pelanggan (konsol)

1. Masuk ke AWS Management Console 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 Kelola 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, memberi, atau menghapus tanda untuk kebijakan terkelola pelanggan. Anda dapat menggunakan AWS CLI atau AWS API untuk mengelola tanda bagi kebijakan terkelola pelanggan.

Untuk membuat daftar tag yang saat ini ada pada kebijakan terkelola pelanggan (AWS CLI atau AWS API)

- AWS CLI: [AWS list-policy-tags](#)
- AWS API: [ListPolicyTags](#)

Untuk melampirkan tanda ke kebijakan terkelola pelanggan (AWS CLI atau AWS API)

- AWS CLI: [aws iam tag-policy](#)
- AWS API: [TagPolicy](#)

Untuk menghapus tanda dari kebijakan terkelola pelanggan (AWS CLI atau AWS API)

- AWS CLI: [aws iam untag-policy](#)
- AWS API: [UntagPolicy](#)

Untuk informasi tentang memberi tanda ke sumber daya untuk layanan AWS lainnya, 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 IAM

Anda dapat menggunakan pasangan nilai kunci tag IAM untuk menambahkan atribut khusus ke penyedia identitas IAM (). IdPs

Anda juga dapat menggunakan tag di AWS STS untuk menambahkan atribut khusus saat Anda mengasumsikan peran atau memfederasi pengguna. Untuk informasi selengkapnya, lihat [Melewati tag sesi di AWS STS](#).

Untuk mempelajari tentang penandaan IdPs di IAM, lihat topik berikut:

Topik

- [Menandai penyedia identitas OpenID Connect \(OIDC\)](#)
- [Menandai penyedia identitas IAM SAM](#)

Menandai penyedia identitas OpenID Connect (OIDC)

Anda dapat menggunakan nilai kunci IAM tag untuk menambahkan atribut kustom 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 yang diperlukan untuk menandai penyedia identitas IAM OIDC

Anda harus mengonfigurasi izin untuk mengizinkan entitas IAM OIDC (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 OIDC (pengguna atau peran) untuk menambahkan, membuat daftar, atau menghapus tanda bagi 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 tandanya 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) untuk menambahkan tanda ke penyedia identitas IAM OIDC tertentu

Tambahkan pernyataan berikut ini ke kebijakan izin untuk entitas IAM yang perlu menambahkan, tapi tidak menghapus, tanda 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 tandanya 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>"
}
```

```
}
```

Cara alternatif, Anda dapat menggunakan AWS kebijakan terkelola seperti [IAMFullAccess](#) untuk menyediakan akses penuh ke IAM.

Mengelola tanda pada penyedia identitas IAM OIDC (konsol)

Anda dapat mengelola tanda untuk penyedia identitas IAM OIDC dari AWS Management Console.

 Note

Anda dapat mengelola tanda hanya dengan menggunakan pengalaman konsol penyedia identitas baru.

Untuk mengelola tanda pada penyedia identitas OIDC (konsol)

1. Masuk ke AWS Management Console 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 bagian Tanda, pilih Kelola tanda, kemudian 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 tanda pada penyedia identitas IAM OIDC (AWS CLI atau AWS API)

Anda dapat membuat daftar, melampirkan, atau menghapus tanda untuk penyedia identitas IAM OIDC. Anda dapat menggunakan AWS CLI atau AWS API untuk mengelola tag untuk penyedia identitas IAM OIDC.

Untuk membuat daftar tanda saat ini ada pada penyedia identitas IAM OIDC (AWS CLI atau AWS API)

- AWS CLI: [aws list-open-id-connect-Provider tag](#)
- API AWS: [ListOpenIDConnectProviderTags](#)

Untuk melampirkan tanda ke penyedia identitas IAM OIDC (AWS CLI atau AWS API)

- AWS CLI: [aws tag-open-id-connectpenyedia](#)
- API AWS: [TagOpenIDConnectProvider](#)

Untuk menghapus tanda dari penyedia identitas IAM OIDC (AWS CLI atau AWS API)

- AWS CLI: [aws untag-open-id-connectpenyedia](#)
- API AWS: [UntagOpenIDConnectProvider](#)

Untuk informasi tentang memberi tanda ke sumber daya untuk layanan AWS lainnya, 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 IAM SAM

Anda dapat menggunakan pasangan kunci-nilai tanda IAM untuk menambahkan atribut kustom 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 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) guna menambahkan, membuat daftar, atau menghapus tanda bagi penyedia identitas SAML

Tambahkan pernyataan berikut ini ke kebijakan izin untuk entitas IAM yang perlu mengelola tag. Gunakan nomor akun Anda dan ganti *ProviderName<SMM >* dengan nama penyedia SMM yang tandanya 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) guna menambahkan tanda ke penyedia identitas SMM tertentu

Tambahkan pernyataan berikut ini ke kebijakan izin untuk entitas IAM yang perlu menambahkan, tapi tidak menghapus, tanda untuk penyedia SMM tertentu.

 Note

Tindakan `iam:TagSAMLProvider` mengharuskan Anda untuk juga menyertakan tindakan `iam:ListSAMLProviderTags`

Untuk menggunakan kebijakan ini, ganti *ProviderName<SMM >* dengan nama penyedia SMM yang tandanya 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 yang AWS dikelola oleh [IAMFullAccess](#) untuk memberikan akses penuh ke IAM.

Mengelola tanda pada penyedia identitas IAM SAM (konsol)

Anda dapat mengelola tanda untuk Penyedia IAM SAM dari AWS Management Console.

 Note

Anda dapat mengelola tanda hanya dengan menggunakan pengalaman konsol penyedia identitas baru.

Untuk mengelola tanda pada penyedia identitas SAML (konsol)

1. Masuk ke AWS Management Console 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 bagian Tanda, pilih Kelola tanda, kemudian 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 SALL (AWS CLI atau AWS API)

Anda dapat membuat daftar, melampirkan, atau menghapus tanda untuk penyedia identitas IAM SAM. Anda dapat menggunakan AWS CLI atau AWS API untuk mengelola tanda untuk penyedia identitas IAM SAM.

Untuk membuat daftar tanda yang saat ini dilampirkan ke penyedia identitas SAML (AWS CLI atau API AWS)

- AWS CLI: [AWS list-saml-provider-tags](#)

- API AWS: [ListSamlProviderTags](#)

Untuk melampirkan tanda ke penyedia identitas SAML (AWS CLI atau API AWS)

- AWS CLI: [AWS tag-saml-provider](#)
- API AWS: [TagsAMLProvider](#)

Untuk menghapus tanda dari penyedia identitas SAML (AWS CLI atau API AWS)

- AWS CLI: [AWS untag-saml-provider](#)
- API AWS: [UntagsAMLProvider](#)

Untuk informasi tentang memberi tanda ke sumber daya untuk layanan AWS lainnya, 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 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 instans ketika menggunakan AWS CLI atau API AWS.

Anda dapat menggunakan pasangan nilai kunci tanda IAM untuk menambahkan atribut kustom ke profil instans. 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 di AWS STS untuk menambahkan atribut khusus saat Anda mengasumsikan peran atau memfederasi pengguna. Untuk informasi selengkapnya, lihat [Melewati 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 instans. 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) untuk menambahkan, atau menghapus tanda bagi profil instans

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 tandanya 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) untuk menambahkan tanda ke profil instans tertentu

Tambahkan pernyataan berikut ini ke kebijakan izin untuk entitas IAM yang perlu menambahkan, tapi tidak menghapus, tag untuk profil instans tertentu.

Note

Tindakan `iam:TagInstanceProfile` mengharuskan Anda untuk juga menyertakan tindakan `iam:ListInstanceProfileTags`

Untuk menggunakan kebijakan ini, ganti *<InstanceProfileName >* dengan nama profil instans yang tandanya 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 yang AWS dikelola seperti [IAMFullAccess](#) untuk memberikan akses penuh ke IAM.

Mengelola tanda di profil instans (AWS CLI atau API AWS)

Anda dapat membuat daftar, melampirkan, atau menghapus tanda untuk profil instans. Anda dapat menggunakan AWS CLI atau API AWS untuk mengelola tanda bagi profil instans.

Untuk membuat daftar tanda yang saat ini dilampirkan ke profil instans (AWS CLI atau API AWS)

- AWS CLI: [AWS list-instance-profile-tags](#)
- AWSAPI: [ListInstanceProfileTags](#)

Untuk melampirkan tanda ke profil instans (AWS CLI atau API AWS)

- AWS CLI: [AWS tag-instance-profile](#)
- AWSAPI: [TagInstanceProfile](#)

Untuk menghapus tanda dari profil instans (AWS CLI atau API AWS)

- AWS CLI: [AWS untag-instance-profile](#)
- AWSAPI: [UntagInstanceProfile](#)

Untuk informasi tentang memberi tanda ke sumber daya untuk layanan AWS lainnya, 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 sertifikat server

Jika Anda menggunakan IAM untuk mengelola sertifikat SSL/TLS, Anda dapat menandai sertifikat server di IAM menggunakan atau API. AWS CLI AWS Untuk sertifikat di Wilayah yang didukung oleh AWS Certificate Manager (ACM), sebaiknya Anda menggunakan ACM alih-alih IAM untuk menyediakan, mengelola, dan menerapkan 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 kustom saat Anda mengambil peran atau menyatukan pengguna. Untuk informasi selengkapnya, lihat [Melewati tag sesi di AWS STS](#).

Izin yang diperlukan untuk menandai sertifikat server

Anda harus mengonfigurasi izin untuk mengizinkan entitas IAM (pengguna atau peran) 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) untuk 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) untuk 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 [IAM FullAccess](#) 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 kustom saat Anda mengambil peran atau menyatukan pengguna. Untuk informasi selengkapnya, lihat [Melewati tag sesi di AWS STS](#).

Izin yang diperlukan untuk menandai perangkat MFA virtual

Anda harus mengonfigurasi izin untuk mengizinkan entitas IAM (pengguna atau peran) 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) menambahkan, 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 tandanya 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 tandanya 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 [IAM FullAccess](#) 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: [ListMFA Device Tags](#)

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](#).

Melewati tag sesi di AWS STS

Tag sesi adalah atribut pasangan nilai kunci yang Anda lewati saat Anda mengambil peran IAM atau menyatukan pengguna. AWS STS Anda melakukan ini dengan membuat permintaan AWS CLI atau AWS API melalui AWS STS atau melalui penyedia identitas Anda (iDP). Saat Anda menggunakan AWS STS untuk meminta kredensial 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 diteruskan saat Anda mengambil peran atau menggabungkan pengguna menggunakan AWS API AWS CLI atau. 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 ke IAM Anda mengambil peran.

Topik

- [Operasi pemberian tag ke sesi](#)
- [Hal yang perlu diketahui tentang tag sesi](#)

- [Izin diperlukan untuk menambahkan tag sesi](#)
- [Meneruskan tag sesi dengan menggunakan AssumeRole](#)
- [Melewati tag sesi menggunakan AssumeRoleWith SAMP](#)
- [Meneruskan tag sesi dengan menggunakan AssumeRoleWithWebIdentity](#)
- [Meneruskan tag sesi dengan 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 AWS Management Console [Switch Role](#) 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](#).

Membandingkan metode untuk meneruskan tag sesi

Operasi	Siapa yang bisa mengambil peran	Metode untuk melewati 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 atau --transitive-tag-keys Opsi CLI
Operasi assume-role-with-saml CLI atau AssumeRoleWithSAML API	Setiap pengguna yang diautentikasi menggunakan penyedia identitas SAMP	PrincipalTag Atribut SAML	TransitiveTagKeys Atribut SAML

Operasi	Siapa yang bisa mengambil peran	Metode untuk melewati tag	Metode untuk mengatur tag transitif
Operasi assume-role-with-web-identity CLI atau AssumeRoleWithWebIdentity API	Setiap pengguna yang diautentikasi menggunakan penyedia OIDC	Principal Tag Token OIDC	Transitiv eTagKeys 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.
- Plaintext kunci tag sesi Anda melebihi 128 karakter.
- Plaintext dari nilai tag sesi Anda melebihi 256 karakter.
- Ukuran total plaintext kebijakan sesi melebihi 2048 karakter.
- Ukuran paket total dari kebijakan sesi gabungan dan tag terlalu besar. Jika operasi gagal, pesan kesalahan menunjukkan seberapa dekat kebijakan dan tag yang digabungkan mencapai batas ukuran atas, berdasarkan persentase.

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 penerusan penyedia identitas (iDP) harus memiliki izin. [sts:TagSession](#) Untuk peran tanpa izin ini dalam kebijakan kepercayaan, AssumeRole operasi gagal.

- Saat Anda 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 mengganti peran yang diasumsikan atau tag pengguna gabungan yang ada dengan kunci tag yang sama, terlepas dari kasus karakternya.
- Anda tidak dapat meneruskan tag sesi menggunakan AWS Management Console.
- 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 bertahan selama rantai peran dan menggantikan 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 memberikan 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](#).
- AWS Konversi memampatkan kebijakan sesi yang diteruskan dan tag sesi yang digabungkan ke dalam format biner yang dikemas dengan batas terpisah. Jika Anda melebihi batas ini, pesan kesalahan AWS CLI atau AWS API menunjukkan seberapa dekat kebijakan dan tag yang digabungkan mencapai batas ukuran atas, berdasarkan persentase.

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`. `AssumeRoleOperation` gagal untuk peran apa pun yang terhubung ke tag sesi idP yang lewat 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 prinsipal 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 nilai kunci yang dilampirkan ke sumber daya. Misalnya, Anda dapat mengizinkan prinsipal untuk meneruskan tag sesi hanya jika peran yang mereka ambil menyertakan 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 tersebut, 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

```
{
  "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?

- `AllowIamUserAssumeRole` Pernyataan tersebut 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, sehingga 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 kumpulan tag maksimum yang dapat Anda atur 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 transitif dengan menambahkan blok syarat lain yang mencakup `"Null":{"sts:TransitiveTagKeys":"false"}`.

Meneruskan tag sesi dengan menggunakan AssumeRole

AssumeRoleOperasi mengembalikan satu set kredensi sementara yang dapat Anda gunakan untuk mengakses AWS sumber daya. Anda dapat menggunakan kredensi pengguna atau peran IAM untuk menelepon. AssumeRole Untuk meneruskan tag sesi sambil mengambil peran, gunakan `--tags` AWS CLI opsi atau parameter `Tags` AWS API.

Untuk menetapkan 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.

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 AssumeRoleWith SAMP

AssumeRoleWithSAMPOperasi mengotentikasi dengan federasi berbasis SAMP. Operasi ini mengembalikan satu set kredensi sementara yang dapat Anda gunakan untuk mengakses AWS sumber daya. Untuk informasi selengkapnya tentang penggunaan federasi berbasis SAMP untuk AWS Management Console akses, lihat [Mengaktifkan pengguna gabungan SAML 2.0 untuk mengakses AWS Management Console](#) Untuk detail tentang AWS CLI atau akses AWS API, lihat [Federasi SAML 2.0](#). Untuk tutorial tentang mengonfigurasi federasi SAMP untuk pengguna Active Directory Anda, lihat [AWS Otentikasi 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. [Konfigurasi jaringan Anda sebagai penyedia SAMP untuk AWS.](#)
2. [Buat penyedia SAMP di IAM](#)
3. [Konfigurasi peran dan izin AWS untuk pengguna federasi](#)
4. [Selesai mengonfigurasi SAMP iDP dan membuat pernyataan untuk respons otentikasi SAMP](#)

AWS termasuk penyedia identitas dengan end-to-end pengalaman bersertifikat untuk tag sesi dengan solusi identitas mereka. Untuk mempelajari cara menggunakan penyedia identitas ini untuk mengonfigurasi tag sesi, lihat [Integrasikan penyedia solusi SAMP 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 Project dan Department tag 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>
```

Meneruskan tag sesi dengan menggunakan AssumeRoleWithWebIdentity

Gunakan federasi yang sesuai dengan OpenID Connect (OIDC) untuk mengautentikasi operasi. AssumeRoleWithWebIdentity Operasi ini mengembalikan satu set kredensi sementara yang dapat Anda gunakan untuk mengakses AWS sumber daya. Untuk informasi selengkapnya tentang penggunaan federasi identitas web untuk AWS Management Console akses, lihat [Federasi OIDC](#).

Untuk meneruskan tanda sesi dari OpenID Connect (OIDC), Anda harus menyertakan tanda sesi dalam JSON Web Token (JWT). Sertakan tag sesi dalam namespace <https://aws.amazon.com/> tags 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 didekodekan berikut menggunakan token untuk memanggil AssumeRoleWithWebIdentity dengan tagProject, CostCenter, dan sesi. Department Token juga mengatur Project dan CostCenter tag sebagai transitif. Tag transitif tetap ada selama perangkaian peran. Untuk informasi selengkapnya, lihat [Merangkai peran dengan tag sesi](#).

Example Contoh Token Web JSON yang diterjemahkan

```
{
  "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"
      ]
    }
  }
}

```

Meneruskan tag sesi dengan menggunakan GetFederationToken

`GetFederationToken` ini memungkinkan Anda untuk menyatukan pengguna Anda. Operasi ini mengembalikan satu set kredensi 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 menyetel tag sesi sebagai transitif saat Anda 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 kredensi sementara yang dikembalikan oleh `GetFederationToken` operasi, tag utama sesi menyertakan tag pengguna dan tag sesi yang diteruskan.

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 bertahan selama rantai peran dan menggantikan ResourceTag nilai yang cocok setelah evaluasi kebijakan kepercayaan peran.

Contoh berikut menunjukkan bagaimana AWS STS meneruskan 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.

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.

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 kredensial 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 dan mewarisi tag sesi transitif di sesi pertama. Sesi kedua yang dihasilkan tag utama adalah Heart =1, Star =1, dan Sun =2. Heart dan Star terus menjadi transitif. SunTag yang Role2 dilampirkan tidak ditandai sebagai transitif karena bukan tag sesi. Sesi mendatang tidak mewarisi tag ini.

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 untuk melewati tag 2 sesi Sun =, operasi gagal. Tag sesi Star = 1 yang diwarisi menggantikan tag role Star =. 3 Dalam rantai peran, nilai tag transitif mengesampingkan peran yang cocok dengan ResourceTag nilai setelah evaluasi kebijakan kepercayaan peran. Dalam contoh ini, jika Role3 menggunakan Star sebagai ResourceTag dalam kebijakan kepercayaan peran, dan menetapkan ResourceTag nilai ke nilai tag transitif dari sesi peran pemanggilan. LightningTag peran juga berlaku untuk sesi ketiga, dan tidak ditetapkan sebagai transitif.

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 access control (ABAC) menggunakan strategi otorisasi yang mendefinisikan izin berdasarkan atribut tag.

Jika perusahaan menggunakan penyedia identitas (IdP) berbasis OIDC atau SAML untuk mengelola identitas pengguna, Anda dapat mengonfigurasi pernyataan untuk meneruskan tanda sesi ke AWS.

Misalnya, dengan identitas pengguna perusahaan, ketika karyawan Anda bergabung AWS, AWS menerapkan atribut mereka pada prinsipal yang dihasilkan. Kemudian, Anda dapat menggunakan ABAC untuk mengizinkan atau menolak izin berdasarkan atribut tersebut. Untuk detailnya, lihat [Tutorial IAM: Menggunakan tanda sesi SAML untuk ABAC](#).

Untuk informasi selengkapnya tentang penggunaan Pusat Identitas IAM dengan ABAC, lihat [Atribut untuk kontrol akses](#) di AWS IAM Identity Center Panduan Pengguna.

Melihat tag sesi di CloudTrail

Anda dapat menggunakan AWS CloudTrail untuk melihat permintaan yang digunakan untuk mengambil peran atau pengguna federasi. File log CloudTrail mencakup informasi tentang tanda prinsipal untuk sesi peran yang diasumsikan atau sesi pengguna gabungan. Untuk informasi selengkapnya, lihat [Mencatat panggilan IAM dan AWS STS 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 AssumeRoleWith log SAMP CloudTrail

```
"requestParameters": {
  "sAMLAAssertionID": "_c0046cEXAMPLEb9d4b8eEXAMPLE2619aEXAMPLE",
  "roleSessionName": "MyRoleSessionName",
  "principalTags": {
    "CostCenter": "987654",
    "Project": "Unicorn"
  },
  "transitiveTagKeys": [
    "CostCenter",
    "Project"
  ],
  "durationSeconds": 3600,
  "roleArn": "arn:aws:iam::123456789012:role/SAMLTTestRoleShibboleth",
  "principalArn": "arn:aws:iam::123456789012:saml-provider/Shibboleth"
},
```

Anda dapat melihat contoh CloudTrail log berikut untuk melihat peristiwa yang menggunakan tag sesi.

- [Contoh peristiwa API rantai AWS STS peran dalam file CloudTrail log](#)

- [Contoh peristiwa SALL AWS STS API dalam file CloudTrail log](#)
- [Contoh peristiwa AWS STS API OIDC dalam file log CloudTrail](#)

Mencatat panggilan IAM dan AWS STS 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 AWS STS sebagai peristiwa, 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 untuk 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 selengkapnya CloudTrail, lihat [Panduan AWS CloudTrail Pengguna](#).

Topik

- [IAM dan AWS STS informasi di CloudTrail](#)
- [Pencatatan permintaan IAM dan AWS STS API](#)
- [Mencatat permintaan API ke layanan AWS lainnya](#)
- [Mencatat peristiwa masuk pengguna](#)
- [Mencatat peristiwa masuk untuk kredensial sementara](#)
- [Contoh peristiwa API IAM dalam catatan CloudTrail](#)
- [Contoh peristiwa AWS STS API di CloudTrail log](#)
- [Contoh peristiwa masuk dalam catatan CloudTrail](#)
- [Perilaku kebijakan kepercayaan peran IAM](#)

IAM dan AWS STS informasi di CloudTrail

CloudTrail diaktifkan pada 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 dalam riwayat Acara. Anda dapat melihat, mencari, dan mengunduh acara

terbaru di situs Anda Akun AWS. Untuk informasi selengkapnya, lihat [Melihat Acara dengan Riwayat CloudTrail Acara](#).

Untuk catatan acara yang sedang berlangsung di Anda Akun AWS, termasuk acara untuk IAM dan AWS STS, buat jejak. Jejak memungkinkan CloudTrail untuk mengirimkan 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 menindaklanjuti data peristiwa yang dikumpulkan dalam CloudTrail log. Lihat informasi yang lebih lengkap di:

- [Ikhtisar Pembuatan Jejak](#)
- [CloudTrail Layanan dan Integrasi yang Didukung](#)
- [Mengonfigurasi Notifikasi Amazon SNS untuk CloudTrail](#)
- [Menerima File CloudTrail Log dari Beberapa Wilayah](#) dan [Menerima File CloudTrail 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 permintaan IAM dan AWS STS API

CloudTrail mencatat semua permintaan API yang diautentikasi (dibuat dengan kredensial) ke operasi IAM dan API. AWS STS CloudTrail juga mencatat permintaan yang tidak diautentikasi ke AWS STS tindakan, `AssumeRoleWithSAML` dan `AssumeRoleWithWebIdentity`, dan mencatat informasi yang diberikan oleh penyedia identitas. Anda dapat menggunakan informasi ini untuk memetakan panggilan yang dilakukan oleh pengguna gabungan dengan peran yang diasumsikan kembali ke pemanggil gabungan eksternal asal. Dalam hal ini `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 butuhkan untuk memetakan permintaan `AssumeRole` * dengan pengguna federasi tertentu. Untuk informasi selengkapnya, lihat [Elemen CloudTrail UserIdentity](#) di AWS CloudTrail Panduan Pengguna.

Misalnya, panggilan ke `IAMCreateUser`, `DeleteRoleListGroups`, dan operasi API lainnya semuanya dicatat oleh CloudTrail.

Contoh untuk jenis entri log ini akan disampaikan nanti dalam topik ini.

Mencatat permintaan API ke layanan AWS lainnya

Permintaan yang diautentikasi ke operasi API AWS layanan lainnya dicatat oleh CloudTrail, dan entri log ini berisi informasi tentang siapa yang membuat permintaan tersebut.

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 di entri CloudTrail log, lihat Elemen [UserIdentity](#) di AWS CloudTrail Panduan Pengguna.

Mencatat peristiwa masuk pengguna

CloudTrail mencatat peristiwa masuk ke AWS Management Console, forum AWS diskusi, dan AWS Marketplace. CloudTrail mencatat upaya masuk yang berhasil dan gagal untuk pengguna IAM dan pengguna federasi.

Untuk melihat contoh CloudTrail peristiwa untuk login pengguna root yang berhasil dan tidak berhasil, lihat [Contoh catatan peristiwa untuk pengguna root di Panduan Pengguna](#). AWS CloudTrail

Sebagai praktik keamanan terbaik, 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 satunya Akun AWS, tetapi kemudian ketik nomor akun untuk yang berbeda Akun AWS.
- Anda lupa akun mana yang sedang Anda masuki dan secara tidak sengaja mengetikkan nama akun dari akun email pribadi Anda, pengenalan 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, prinsipal memanggil AWS STS API untuk mengambil kredensial sementara. Operasi itu masuk ke 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 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 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 kepala sekolah yang berbeda saat Anda meninjau log. AWS CloudTrail

Tabel berikut menunjukkan cara CloudTrail mencatat 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 kredensialnya	GetSessionToken	Identitas pengguna root	Akun pemilik peran sama dengan akun panggilan	Identitas pengguna root
Pengguna IAM	GetSessionToken	Identitas pengguna IAM	Akun pemilik peran sama	Identitas pengguna IAM

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
			dengan akun panggilan	
Pengguna IAM	GetFederationToken	Identitas pengguna IAM	Akun pemilik peran sama dengan akun panggilan	Identitas pengguna IAM
Pengguna IAM	AssumeRole	Identitas pengguna IAM	Nomor rekening dan ID utama (jika pengguna), atau kepala AWS layanan	Khusus identitas peran (tanpa pengguna)
Pengguna yang diautentikasi secara eksternal	AssumeRoleWithSAML	tidak ada	Identitas pengguna SAML	Khusus identitas peran (tanpa pengguna)
Pengguna yang diautentikasi secara eksternal	AssumeRoleWithWebIdentity	tidak ada	Identitas pengguna OIDC/Web	Khusus identitas peran (tanpa pengguna)

CloudTrail menganggap tindakan hanya-baca jika tidak memiliki efek mutasi pada sumber daya. Saat mencatat peristiwa hanya-baca, CloudTrail menyunting `responseElements` informasi di log. Saat CloudTrail mencatat peristiwa yang tidak hanya-baca, lengkap `responseElements` ditampilkan di entri log. Namun, untuk AWS STS API `AssumeRole`, `AssumeRoleWithSAML`, dan `AssumeRoleWithWebIdentity`, meskipun dicatat sebagai hanya-baca, CloudTrail akan menyertakan lengkap `responseElements` dalam log untuk API ini.

Tabel berikut menunjukkan bagaimana CloudTrail log `responseElements` dan `readOnly` informasi untuk setiap AWS STS API yang menghasilkan kredensi sementara.

API STS	Informasi elemen respons	Hanya baca
AssumeRole	Termasuk	true
AssumeRoleWithSAM	Termasuk	true
AssumeRoleWithWebIdentity	Termasuk	true
GetFederationToken	Termasuk	false
GetSessionToken	Termasuk	false

Contoh peristiwa API IAM dalam catatan 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 API IAM dalam file CloudTrail log

Contoh berikut menunjukkan entri CloudTrail log untuk permintaan yang dibuat untuk `GetUserPolicy` tindakan IAM.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn": "arn:aws:iam::444455556666:user/JaneDoe",
    "accountId": "444455556666",
    "accessKeyId": "AKIAI44QH8DHBEXAMPLE",
    "userName": "JaneDoe",
    "sessionContext": {
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2014-07-15T21:39:40Z"
      }
    }
  },
  "invokedBy": "signin.amazonaws.com"
},
"eventTime": "2014-07-15T21:40:14Z",
```

```
"eventSource": "iam.amazonaws.com",
"eventName": "GetUserPolicy",
"awsRegion": "us-east-2",
"sourceIPAddress": "signin.amazonaws.com",
"userAgent": "signin.amazonaws.com",
"requestParameters": {
  "userName": "JaneDoe",
  "policyName": "ReadOnlyAccess-JaneDoe-201407151307"
},
"responseElements": null,
"requestID": "9EXAMPLE-0c68-11e4-a24e-d5e16EXAMPLE",
"eventID": "cEXAMPLE-127e-4632-980d-505a4EXAMPLE"
}
```

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 AWS Management Console, seperti yang Anda tahu dari `userAgent` elemen.

Contoh peristiwa AWS STS API di 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 peristiwa AWS STS API lintas akun dalam file CloudTrail log

Pengguna IAM yang disebutkan `JohnDoe` di akun `777788889999` memanggil AWS STS `AssumeRole` tindakan untuk mengambil peran dalam 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 `JohnDoe`.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDAQRSTUVWXYZEXAMPLE",
    "arn": "arn:aws:iam::777788889999:user/JohnDoe",
    "accountId": "777788889999",
```

```
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "userName": "JohnDoe"
  },
  "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 boto-core/1.4.67",
  "requestParameters": {
    "roleArn": "arn:aws:iam::111122223333:role/EC2-dev",
    "roleSessionName": "JohnDoe-EC2-dev",
    "sourceIdentity": "JohnDoe",
    "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:JohnDoe-EC2-dev",
      "arn": "arn:aws:sts::111122223333:assumed-role/EC2-dev/JohnDoe-EC2-dev"
    },
    "sourceIdentity": "JohnDoe"
  },
  "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": "JohnDoe-EC2-dev",
    "sourceIdentity": "JohnDoe",
    "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:JohnDoe-EC2-dev",
      "arn": "arn:aws:sts::111122223333:assumed-role/EC2-dev/JohnDoe-EC2-dev"
    },
    "sourceIdentity": "JohnDoe"
  },
  "requestID": "4EXAMPLE-0e8d-11e4-96e4-e55c0EXAMPLE",
  "sharedEventID": "bEXAMPLE-efea-4a70-b951-19a88EXAMPLE",
  "eventID": "dEXAMPLE-ac7f-466c-a608-4ac8dEXAMPLE"
}
```

Contoh peristiwa API rantai AWS STS 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 JohnDoe 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 JohnDoe1 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/JohnDoe/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": "JohnDoe"
      },
      "sourceIdentity": "JohnDoe"
    }
  },
  "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 boto3/1.12.239",
"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": "JohnDoe",
  "transitiveTagKeys": [
    "Email",
    "CostCenter"
  ],
  "durationSeconds": 3600
},
"responseElements": {
  "credentials": {
    "accessKeyId": "ASIAI44QH8DHBEXAMPLE",
    "expiration": "Oct 2, 2019, 11:12:29 PM",
    "sessionToken": "AgoJb3JpZ2luX2VjEB4aCXVzLXdlc3QtMSJHMEXAMPLETOKEN
+//rJb8Lo30mFc5M1hFCEbubZvEj0wHB/mDMwIgSEe9gk/Zjr09tZV7F1HDTMhmEXAMPLETOKEN/iEJ/
rkqngII9//////////
ARABGgw0MjgzMdc4NjM5NjYiDLZjZFKwP4qxQG5sFCryAS04UPz5qE97wPPH1eLMvs7CgSDBSwfonmRTCfokm2FN1+hWUdQ
+C+WKFZb701eiv9J5La2EXAMPLETOKEN/c7S5Iro1WUJ0q3Cxuo/8HUoSxVhQHM7zF7mWWLhXLEQ52ivL
+F6q5dpXu4aTFedpMfnJa8JtkWwG9x1Axj0Ypy2ok8v5unpQGWych1vwdvj6ez1Dm8Xg1+qIzXILiEXAMPLETOKEN/
vQGqu8H+nxp3kabcrt0vTFTvxX6vsc80GwUfHhZAfYGGEXAMPLETOKEN/
L6v1yMM3B10wF0rQBno1HEjff1oNI8RnQiMNFdU0twYj7HUZIOCMjfn8PPHq77N7GJ191zvIZKQA00wcjg
+mc78zHCj8y0siY8C96paEXAMPLETOKEN/
E3cpksxWdgs91HRzJWScjN2+r2LTGjYhyPqcmFzZo2mCE7mBNEXAMPLETOKEN/oJy
+2o83YNW5t0iDmzczgDzJZ4UKR84yGYOMfSnF4XcEJrDgAJ30JFwmTcTQICALSwLEXAMPLETOKEN"
  },
  "assumedRoleUser": {

```

```

        "assumedRoleId": "AROAIFR7WHDTSOYQYHFUE:Role2WithTags",
        "arn": "arn:aws:sts::111111111111:assumed-role/test-role/Role2WithTags"
    },
    "sourceIdentity": "JohnDoe"
},
"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 acara AWS STS API AWS layanan 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 boto-core/1.4.67]",
"requestParameters": {
  "bucketName": "my-test-bucket-cross-account"
},
"responseElements": null,
"requestID": "EXAMPLE463D56D4C",
"eventID": "dEXAMPLE-265a-41e0-9352-4401bEXAMPLE",
"eventType": "AwsApiCall",
"recipientAccountId": "777788889999"
}

```

Contoh peristiwa SALL AWS STS 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 opsionalDurationSeconds, direpresentasikan seperti durationSeconds dalam CloudTrail log, dan disetel ke 1800 detik. Permintaan ini juga mencakup atribut SAFLsourceIdentity, yang diteruskan dalam pernyataan SAFL. 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+ExampleL/jEvs=:SamlExample",
    "userName": "SamlExample",
    "identityProvider": "bdG0nTesti4+ExampleL/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": 1,
      "subject": "SamlExample",
      "subjectType": "transient",
      "issuer": "https://server.example.com/idp/shibboleth",
      "audience": "https://signin.aws.amazon.com/saml",
      "nameQualifier": "bdG0nTesti4+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/SAMLTTestRoleShibboleth"
  },
  {
    "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 peristiwa AWS STS API OIDC dalam file log CloudTrail

Contoh berikut menunjukkan entri CloudTrail log untuk permintaan yang dibuat untuk AWS STS AssumeRoleWithWebIdentity tindakan tersebut. Permintaan mencakup atribut CostCenter dan Project yang melewati token penyedia identitas sebagai [tanda 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.

```

{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "WebIdentityUser",
    "principalId": "accounts.google.com:<id-of-application>.apps.googleusercontent.com:<id-of-user>",
    "userName": "<id of user>",
    "identityProvider": "accounts.google.com"
  },

```

```
"eventTime": "2016-03-23T01:39:51Z",
"eventSource": "sts.amazonaws.com",
"eventName": "AssumeRoleWithWebIdentity",
"awsRegion": "us-east-2",
"sourceIPAddress": "192.0.2.101",
"userAgent": "aws-cli/1.3.23 Python/2.7.6 Linux/2.6.18-164.el5",
"requestParameters": {
  "sourceIdentity": "MyWebIdentityUser",
  "durationSeconds": 3600,
  "roleArn": "arn:aws:iam::444455556666:role/FederatedWebIdentityRole",
  "roleSessionName": "MyAssignedRoleSessionName"
    "principalTags": {
      "CostCenter": "24680",
      "Project": "Pegasus"
    },
    "transitiveTagKeys": [
      "CostCenter",
      "Project"
    ],
  },
"responseElements": {
  "provider": "accounts.google.com",
  "subjectFromWebIdentityToken": "<id of user>",
  "sourceIdentity": "MyWebIdentityUser",
  "audience": "<id of application>.apps.googleusercontent.com",
  "credentials": {
    "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
    "expiration": "Mar 23, 2016, 2:39:51 AM",
    "sessionToken": "<encoded session token blob>"
  },
  "assumedRoleUser": {
    "assumedRoleId": "AROACQRSTUVWRAOEXAMPLE:MyAssignedRoleSessionName",
    "arn": "arn:aws:sts::444455556666:assumed-role/FederatedWebIdentityRole/MyAssignedRoleSessionName"
  }
},
"resources": [
  {
    "ARN": "arn:aws:iam::444455556666:role/FederatedWebIdentityRole",
    "accountId": "444455556666",
    "type": "AWS::IAM::Role"
  }
],
"requestID": "6EXAMPLE-e595-11e5-b2c7-c974fEXAMPLE",
```

```
"eventID": "bEXAMPLE-0b30-4246-b28c-e3da3EXAMPLE",
"eventType": "AwsApiCall",
"recipientAccountId": "444455556666"
}
```

Contoh peristiwa masuk dalam catatan 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 peristiwa sukses masuk dalam file catatan CloudTrail

Contoh berikut menunjukkan entri CloudTrail log untuk acara login yang berhasil.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn": "arn:aws:iam::111122223333:user/JohnDoe",
    "accountId": "111122223333",
    "userName": "JohnDoe"
  },
  "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"
}
```

Untuk detail selengkapnya tentang informasi yang terkandung dalam file CloudTrail log, lihat [Referensi CloudTrail Acara](#) di Panduan AWS CloudTrail Pengguna.

Contoh peristiwa gagal masuk dalam file catatan CloudTrail

Contoh berikut menunjukkan entri CloudTrail log untuk peristiwa login 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 login yang gagal yang disebabkan oleh pengguna memasukkan nama pengguna yang salah. AWS menutupi `userName` teks dengan `HIDDEN_DUE_TO_SECURITY_REASONS` untuk membantu mencegah mengekspos 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 21 September 2022, AWS membuat perubahan pada perilaku kebijakan kepercayaan peran IAM untuk meminta izin eksplisit dalam kebijakan kepercayaan peran ketika suatu

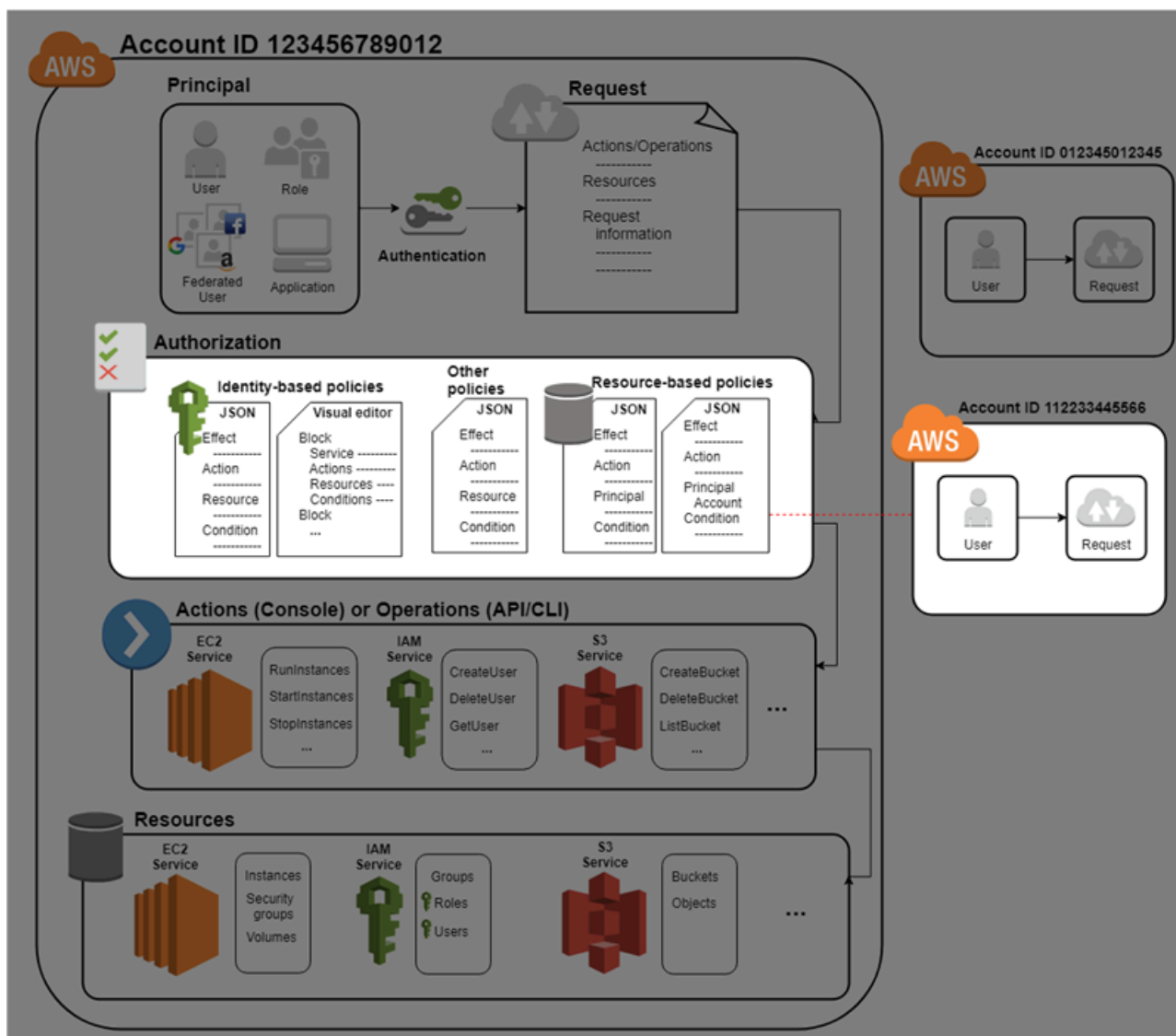
peran mengasumsikan dirinya sendiri. Peran IAM dalam daftar izin perilaku lama memiliki `additionalEventData` bidang yang ada `explicitTrustGrant` untuk `AssumeRole` acara. Nilai `false` ketika peran pada daftar izin lama mengasumsikan dirinya menggunakan perilaku lama. `explicitTrustGrant` Ketika peran pada daftar izin lama mengasumsikan dirinya sendiri tetapi perilaku kebijakan kepercayaan peran telah diperbarui untuk secara eksplisit memungkinkan peran tersebut untuk mengambil alih dirinya sendiri, nilainya benar. `explicitTrustGrant`

Hanya sejumlah kecil peran IAM yang ada di daftar izinkan untuk perilaku lama, dan bidang ini hanya ada di `CloudTrail` log untuk peran ini ketika mereka mengasumsikan diri mereka sendiri. Dalam kebanyakan kasus, peran IAM tidak perlu mengasumsikan 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 pada perilaku kebijakan kepercayaan peran IAM](#).

Manajemen akses untuk sumber daya AWS

AWS Identity and Access Management(IAM) adalah layanan web yang membantu Anda mengendalikan akses keAWS sumber daya secara aman. Saat [penanggung jawab](#) mengajukan permintaan di AWS, kode penegakan AWS memeriksa apakah penanggung jawab diautentikasi (ditandatangani) dan diotorisasi (memiliki izin). Anda mengelola akses diAWS dengan membuat kebijakan dan melampirkannya ke identitas IAM atauAWS sumber daya. Kebijakan adalah dokumen JSON dalam AWS yang, saat dilampirkan ke identitas atau sumber daya, menentukan izinnya. Untuk informasi selengkapnya tentang jenis dan penggunaan kebijakan, lihat [Kebijakan dan Izin di IAM](#).

Untuk perincian tentang proses autentikasi dan otorisasi lainnya, lihat [Cara kerja IAM](#).



Selama otorisasi, kode penegakan AWS menggunakan nilai dari [konteks permintaan](#) untuk memeriksa kesesuaian kebijakan dan menentukan apakah akan mengizinkan atau menolak permintaan tersebut.

AWS memeriksa setiap kebijakan yang berlaku untuk konteks permintaan. Jika kebijakan tunggal 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 ada batasan izin, Organisasi SCP, atau kebijakan sesi ada, hal ini dapat mengesampingkan izin dengan penolakan implisit.
- Penolakan secara tegas dalam kebijakan apa pun akan menimpa izin apa pun.

Setelah permintaan Anda diautentikasi dan diotorisasi, 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.

Sumber daya manajemen akses

Untuk informasi selengkapnya tentang izin dan tentang pembuatan kebijakan, lihat sumber daya berikut:

Entri berikut dalam [BlogAWS Keamanan](#) mencakup berbagai cara umum untuk menulis kebijakan untuk mengakses bucket dan objek Amazon S3.

- [Kebijakan IAM Tulis: Cara Memberikan Akses ke Bucket Amazon S3](#)
- [Kebijakan IAM Tulis: Berikan Akses ke Folder Khusus Pengguna di Bucket Amazon S3](#)
- [Kebijakan IAM dan Kebijakan bucket ACL! Astaga! \(Mengontrol Akses ke Sumber Daya S3\)](#)
- [Panduan Dasar tentang Izin Tingkat Sumber Daya RDS](#)

- [Menulis Izin Tingkat Sumber Daya EC2](#)

Kebijakan dan Izin di IAM

Anda mengelola akses AWS dengan membuat kebijakan dan melampirkannya ke identitas IAM (pengguna, grup pengguna, atau peran) atau sumber daya. AWS 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 dapat menentukan permintaan yang diizinkan atau ditolak. Sebagian besar kebijakan disimpan AWS sebagai dokumen JSON. AWS mendukung enam jenis kebijakan: kebijakan berbasis identitas, kebijakan berbasis sumber daya, batas izin, Organizations SCP, ACL, dan kebijakan sesi.

Kebijakan IAM mendefinisikan izin untuk suatu tindakan terlepas dari metode yang Anda gunakan untuk pengoperasiannya. Misalnya, jika kebijakan mengizinkan [GetUser](#) tindakan, maka pengguna dengan kebijakan tersebut bisa mendapatkan informasi pengguna dari AWS Management Console, API 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 kredensial 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 lebih jarang digunakan, tersedia untuk digunakan di AWS. Untuk perincian selengkapnya, lihat bagian di bawah ini untuk setiap jenis kebijakan.

- [Kebijakan berbasis identitas - Lampirkan kebijakan terkelola](#) dan [sebaris](#) ke identitas IAM (pengguna, grup tempat pengguna berada, atau peran). Kebijakan berbasis identitas memberikan izin kepada sebuah identitas.
- [Kebijakan berbasis sumber daya](#) – Lampirkan kebijakan yang sesuai ke sumber daya. Contoh kebijakan berbasis sumber daya yang paling umum adalah kebijakan bucket Amazon S3 dan kebijakan kepercayaan peran IAM. Kebijakan berbasis sumber daya memberikan izin kepada prinsipal yang ditentukan dalam kebijakan. Prinsipal dapat berada di akun yang sama dengan sumber daya atau di akun lain.
- [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.

- [Organizations SCPs](#) — Gunakan kebijakan kontrol AWS Organizations layanan (SCP) untuk menentukan izin maksimum untuk anggota akun organisasi atau unit organisasi (OU). SCP membatasi izin yang diberikan oleh kebijakan berbasis identitas atau kebijakan berbasis sumber daya kepada entitas (pengguna atau peran) dalam akun, tetapi tidak memberikan izin.
- [Access Control List \(ACL\)](#) — 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.
- [Kebijakan sesi](#) — Lulus kebijakan sesi lanjutan saat Anda menggunakan AWS CLI atau AWS API untuk mengambil peran atau pengguna gabungan. 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](#)

Kebijakan berbasis identitas

Kebijakan berbasis identitas adalah dokumen kebijakan JSON yang mengontrol tindakan apa yang dapat dilakukan oleh identitas (pengguna, grup, dan peran), di sumber daya yang mana, dan di bawah ketentuan-ketentuan apa. Kebijakan berbasis identitas dapat dikategorikan lebih lanjut:

- Kebijakan terkelola — Kebijakan berbasis identitas mandiri yang dapat Anda lampirkan ke beberapa pengguna, grup, dan peran di Anda. Akun AWS Ada dua jenis kebijakan terkelola:
 - AWSkebijakan terkelola — Kebijakan terkelola yang dibuat dan dikelola olehAWS.
 - Kebijakan terkelola pelanggan — Kebijakan terkelola yang Anda buat dan kelola di AndaAkun AWS. Kebijakan yang dikelola pelanggan memberikan kontrol yang lebih tepat atas kebijakan Anda dibandingkan kebijakan yang dikelola AWS.
- Kebijakan inline – Kebijakan yang Anda tambahkan secara langsung ke satu pengguna, grup, atau peran. Kebijakan inline mempertahankan one-to-one hubungan yang ketat antara kebijakan dan identitas. Mereka dihapus ketika Anda menghapus identitas.

Untuk mempelajari cara memilih antara kebijakan terkelola dan kebijakan inline, lihat [Memilih antara kebijakan terkelola dan kebijakan inline](#).

Kebijakan berbasis sumber daya

Kebijakan berbasis sumber daya adalah dokumen kebijakan JSON yang Anda lampirkan ke sumber daya seperti bucket Amazon S3. Kebijakan ini memberikan izin prinsipal yang ditentukan untuk melakukan tindakan tertentu pada sumber daya tersebut dan menetapkan di bawah ketentuan yang berlaku. Kebijakan berbasis sumber daya merupakan kebijakan inline. 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 prinsipal lintas akun ke kebijakan berbasis sumber daya hanya setengah dari membangun hubungan kepercayaan. Ketika prinsipal 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 prinsipal dalam akun yang sama, tidak diperlukan kebijakan berbasis identitas tambahan. Untuk petunjuk langkah demi langkah untuk memberikan akses lintaslayanan, lihat [Tutorial IAM: Delegasikan akses 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 prinsipal 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 [Kebijakan berbasis identitas 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?](#).

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. Kebijakan berbasis sumber daya yang menentukan pengguna atau peran sebagai prinsipal tidak dibatasi oleh batas izin. Penolakan secara eksplisit dalam salah satu kebijakan ini membatalkan izin. Untuk informasi lebih lanjut tentang batas izin, lihat [Batas izin untuk entitas IAM](#).

Kebijakan Pengendalian Layanan (SCPs)

AWS Organizations adalah layanan untuk mengelompokkan dan mengelola secara terpusat Akun AWS yang dimiliki bisnis Anda. Jika Anda mengaktifkan semua fitur dalam suatu organisasi, maka Anda dapat menerapkan kebijakan kontrol layanan (SCP) ke setiap atau semua akun Anda. SCP adalah kebijakan JSON yang menentukan izin maksimum untuk sebuah organisasi atau unit organisasional (OU). SCP membatasi izin untuk entitas dalam akun anggota, termasuk setiap Pengguna root akun AWS. Penolakan secara eksplisit dalam salah satu kebijakan ini membatalkan izin.

Untuk informasi selengkapnya tentang Organizations dan SCP, lihat [Cara Kerja SCP](#) dalam 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 mereka 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 selengkapnya tentang ACL, lihat [Gambaran Umum Daftar Kontrol Akses \(ACL\)](#) dalam Panduan Developer Amazon Simple Storage Service.

Kebijakan sesi

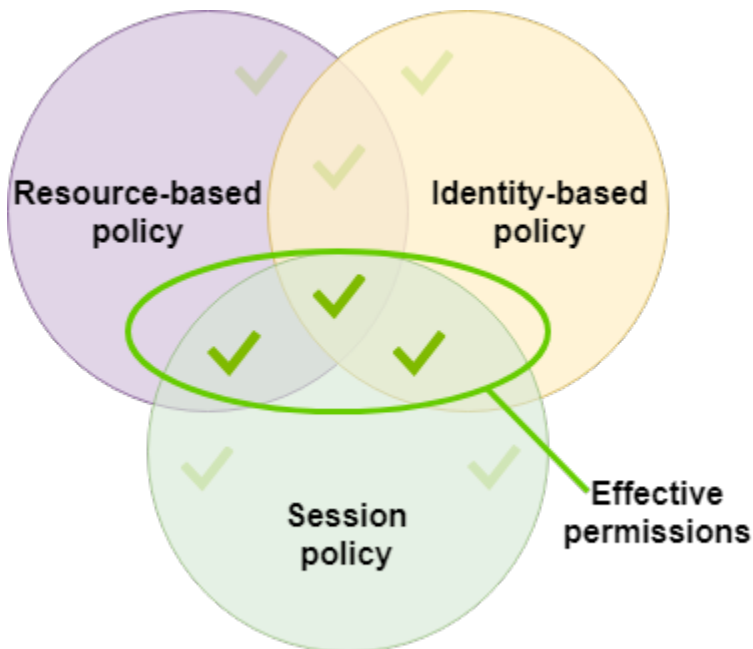
Kebijakan sesi adalah kebijakan lanjutan yang Anda teruskan sebagai parameter saat Anda membuat sesi sementara secara terprogram untuk peran atau pengguna gabungan. Izin untuk sesi adalah titik temu kebijakan berbasis identitas untuk entitas IAM (pengguna atau peran) yang digunakan untuk membuat sesi 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

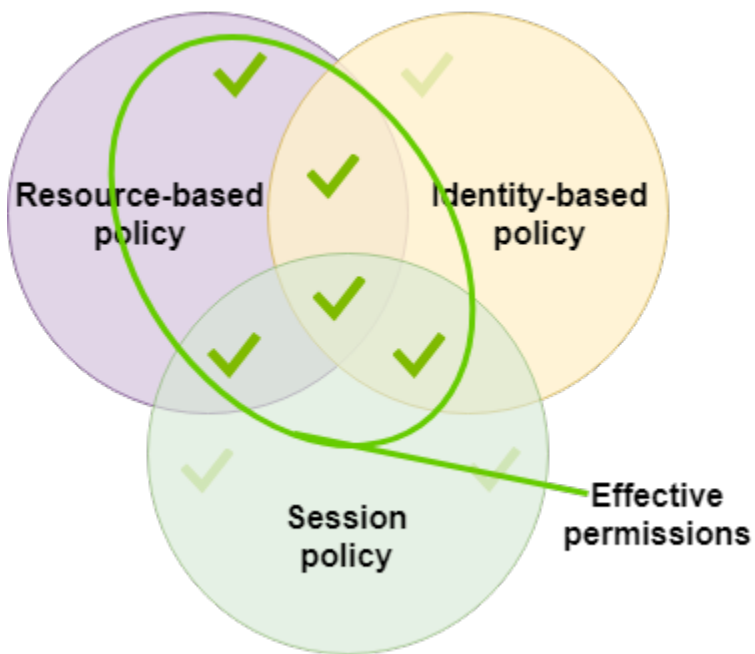
10 kebijakan sesi terkelola. Untuk informasi selengkapnya tentang cara membuat sesi peran, lihat [Meminta kredensial keamanan sementara](#).

Saat membuat sesi 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 [GetFederationToken—federasi melalui pialang identitas khusus](#).

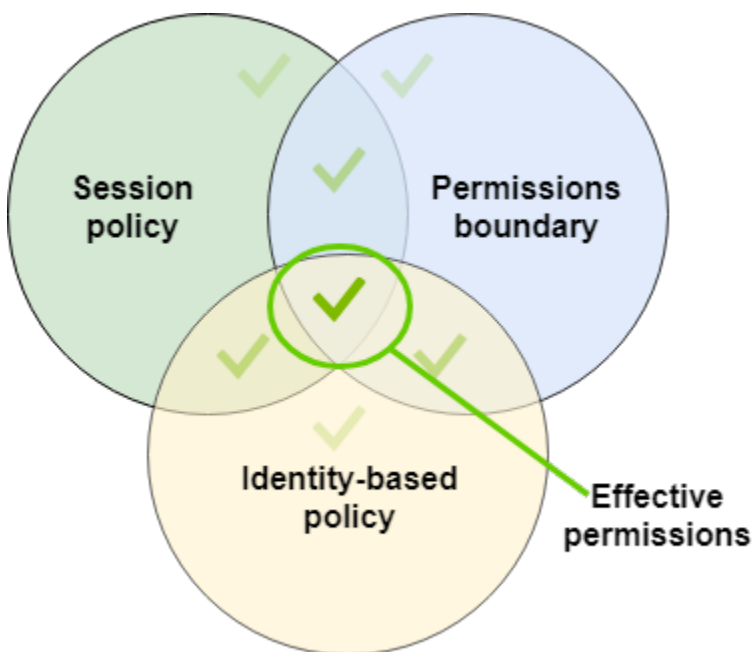
Kebijakan berbasis sumber daya dapat menentukan ARN pengguna atau peran sebagai prinsipal. Dalam hal ini, izin dari kebijakan berbasis sumber daya ditambahkan ke peran atau kebijakan berbasis identitas pengguna sebelum sesi dibuat. Kebijakan sesi membatasi total izin yang diberikan oleh kebijakan berbasis 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

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 apa pun untuk akun tersebut.

Gambaran dari kebijakan JSON

Sebagian besar kebijakan disimpan dalam AWS sebagai dokumen JSON. Kebijakan berbasis identitas dan kebijakan yang digunakan untuk mengatur batas izin adalah dokumen kebijakan JSON yang Anda lampirkan pada pengguna atau peran. Kebijakan berbasis sumber daya adalah dokumen kebijakan JSON yang Anda lampirkan ke sumber daya. SCP adalah dokumen kebijakan JSON dengan sintaks terbatas yang Anda lampirkan ke unit organisasi AWS Organizations. 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 AWS Management Console untuk membuat dan mengubah 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 [Membuat kebijakan IAM](#) dan [Menyunting Kebijakan IAM](#).

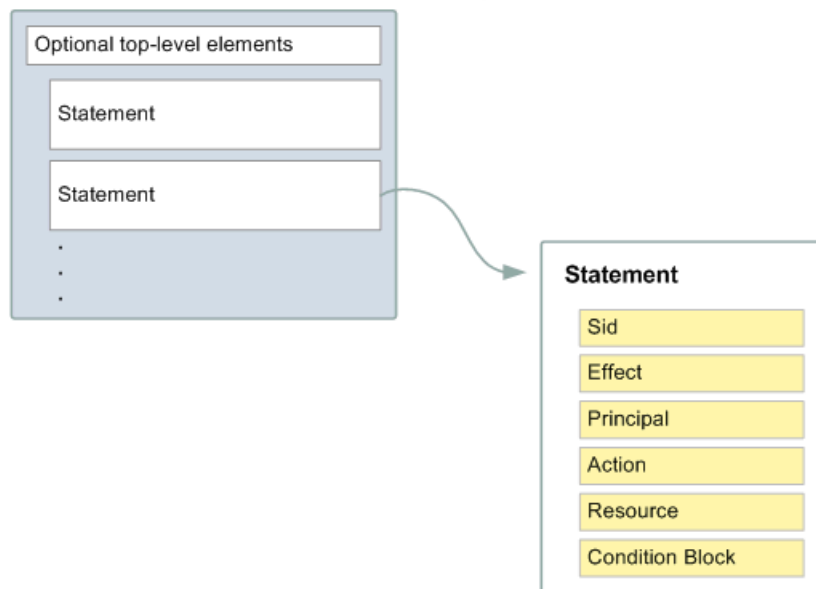
Saat Anda membuat atau mengedit kebijakan JSON, IAM dapat melakukan validasi kebijakan untuk membantu Anda membuat kebijakan yang efektif. IAM mengidentifikasi kesalahan sintaks JSON, sementara IAM Access Analyzer menyediakan pemeriksaan kebijakan tambahan dengan rekomendasi untuk membantu Anda menyempurnakan kebijakan Anda lebih lanjut. Untuk mempelajari selengkapnya tentang validasi kebijakan, lihat [Memvalidasi kebijakan IAM](#). Untuk mempelajari selengkapnya 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 mencakup beberapa pernyataan, AWS menerapkan OR logis di semua pernyataan saat mengevaluasinya. Jika beberapa kebijakan berlaku untuk permintaan, AWS menerapkan OR logis 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 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 (Dijawabkan hanya dalam beberapa situasi)** – Jika Anda membuat kebijakan berbasis sumber daya, Anda harus mengindikasikan akun, pengguna, peran, atau pengguna gabungan yang ingin Anda izinkan atau tolak 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 (Diwajibkan hanya dalam beberapa keadaan) – Jika Anda membuat kebijakan izin IAM, Anda harus menentukan daftar sumber daya ke mana tindakan tersebut berlaku. Jika Anda membuat kebijakan berbasis sumber daya, elemen ini bersifat opsional. Jika Anda tidak menyertakan elemen ini, maka sumber daya yang berlaku adalah sumber daya yang terkait dengan kebijakan tersebut.
- 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. Kami menyarankan Anda memecah 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, AWS [mengevaluasi](#) 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 ember 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 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 kasus ini, Condition mengevaluasi menjadi benar saat pengguna diautentikasi MFA. Jika pengguna tidak diautentikasi MFA, Condition dievaluasi menjadi salah. Dalam hal ini, pernyataan ketiga dalam kebijakan ini tidak berlaku dan pengguna tidak memiliki akses ke bucket confidential-data.

```
{
  "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:::confidential-data",
        "arn:aws:s3:::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 `example_bucket`:

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:ListBucket",
    "Resource": "arn:aws:s3:::example_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 dalam bucket yang diberi nama `mybucket`. 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.)

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "1",
    "Effect": "Allow",
    "Principal": {"AWS": ["arn:aws:iam::account-id:root"]},
    "Action": "s3:*",
    "Resource": [
      "arn:aws:s3:::mybucket",
      "arn:aws:s3:::mybucket/*"
    ]
  }]
}
```

Untuk melihat contoh kebijakan untuk skenario umum, lihat [Contoh kebijakan berbasis identitas IAM](#).

Berikan hak istimewa 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 hanya melakukan 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 [kebijakan atau kebijakan AWS terkelola](#) dengan * izin wildcard untuk memulai kebijakan. Pertimbangkan risiko keamanan memberikan izin kepada kepala sekolah Anda lebih banyak 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 [Memahami tingkat akses dalam ringkasan kebijakan](#).
- Validasi 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 selengkapnya tentang pemeriksaan kebijakan yang disediakan oleh IAM Access Analyzer, lihat validasi kebijakan [IAM Access Analyzer](#).

- Buat kebijakan berdasarkan aktivitas akses — Untuk membantu Anda memperbaiki 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 jangka waktu 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 pengguna atau peran perlu berinteraksi dengan sumber daya AWS untuk kasus penggunaan spesifik Anda. Untuk mempelajari informasi lebih lanjut, lihat [Hasilkan kebijakan berdasarkan aktivitas akses](#).
- Gunakan informasi yang diakses terakhir - Fitur lain yang dapat membantu dengan hak istimewa paling sedikit adalah informasi yang diakses terakhir. Lihat informasi ini di tab Penasihat Akses pada halaman detail konsol IAM untuk pengguna, grup, peran, atau kebijakan IAM. Informasi yang diakses terakhir juga mencakup informasi tentang tindakan yang terakhir diakses untuk beberapa layanan, seperti Amazon EC2, IAM, Lambda, dan Amazon S3. Jika Anda masuk menggunakan kredensial 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 untuk informasi yang terakhir diakses untuk entitas atau kebijakan di IAM atau Organizations. Anda dapat menggunakan informasi ini untuk mengidentifikasi izin yang tidak perlu, sehingga Anda dapat menyempurnakan kebijakan IAM atau Organisasi Anda agar lebih mematuhi prinsip hak istimewa terkecil. Untuk informasi selengkapnya, lihat [Menyaring izin di AWS menggunakan informasi yang terakhir diakses](#).
- Tinjau peristiwa akun di AWS CloudTrail — Untuk mengurangi izin lebih lanjut, Anda dapat melihat peristiwa akun Anda di Riwayat AWS CloudTrail acara. CloudTrail log peristiwa mencakup 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 [Melihat CloudTrail 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.

- [Otentikasi dan Kontrol Akses untuk Amazon DynamoDB](#) di Panduan Pengembang Amazon DynamoDB
- [Menggunakan Kebijakan Bucket dan Kebijakan Pengguna](#) di Panduan Pengguna Amazon Simple Storage Service

- [Ikhtisar Access Control List \(ACL\)](#) di Panduan Pengguna Amazon Simple Storage Service

Kebijakan terkelola dan kebijakan inline

Saat menetapkan izin untuk identitas di IAM, Anda harus memutuskan apakah akan menggunakan kebijakan AWS terkelola, kebijakan yang dikelola pelanggan, atau kebijakan sebaris. Topik berikut memberikan informasi lebih lanjut tentang masing-masing jenis kebijakan berbasis identitas dan kapan menggunakannya.

Topik

- [AWSKebijakan yang dikelola](#)
- [Kebijakan yang dikelola pelanggan](#)
- [Kebijakan inline](#)
- [Memilih antara kebijakan terkelola dan kebijakan inline](#)
- [Memulai dengan kebijakan yang dikelola](#)
- [Mengonversi kebijakan inline ke kebijakan terkelola](#)
- [Kebijakan terkelola yang tidak digunakan lagi AWS](#)

AWSKebijakan yang dikelola

Kebijakan terkelola AWS adalah kebijakan mandiri yang dibuat dan dikelola oleh AWS. Kebijakan mandiri berarti bahwa kebijakan tersebut memiliki Amazon Resource Name (ARN) sendiri yang mencakup nama kebijakan. Misalnya, `arn:aws:iam::aws:policy/IAMReadOnlyAccess` adalah kebijakan terkelola AWS. Untuk informasi selengkapnya tentang ARN, lihat [ARN IAM](#) Untuk mengetahui daftar kebijakan AWS terkelolaLayanan AWS, lihat [kebijakan AWS terkelola](#).

AWSKebijakan terkelola memudahkan Anda untuk menetapkan izin yang sesuai kepada pengguna, grup, 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 terkelola AWS. AWS sesekali memperbarui izin yang ditentukan dalam kebijakan terkelola AWS. Saat AWS melakukan ini, pembaruan memengaruhi semua entitas penanggung jawab (pengguna, grup, dan peran) yang terkait dengan kebijakan tersebut. AWS adalah yang paling memungkinkan untuk memperbarui kebijakan terkelola AWS ketika layanan AWS diluncurkan atau panggilan API

baru tersedia untuk layanan yang sudah ada. Misalnya, kebijakan AWS terkelola yang disebut `ReadOnlyAccess` menyediakan akses hanya-baca ke semua AWS layanan dan sumber daya. Saat AWS meluncurkan layanan baru, AWS memperbarui `ReadOnlyAccess` kebijakan untuk menambahkan izin hanya-baca untuk layanan baru. Izin yang diperbarui diterapkan pada semua entitas prinsipal yang kebijakannya dilampirkan.

Kebijakan AWS terkelola akses penuh menentukan izin untuk administrator layanan dengan memberikan akses penuh ke layanan.

- [AmazonDynamoDB FullAccess](#)
- [IAM FullAccess](#)

Kebijakan yang AWS dikelola pengguna daya menyediakan akses penuh ke AWS layanan dan sumber daya, tetapi tidak mengizinkan mengelola pengguna dan grup.

- [AWSCodeCommitPowerUser](#)
- [AWSKeyManagementServicePowerUser](#)

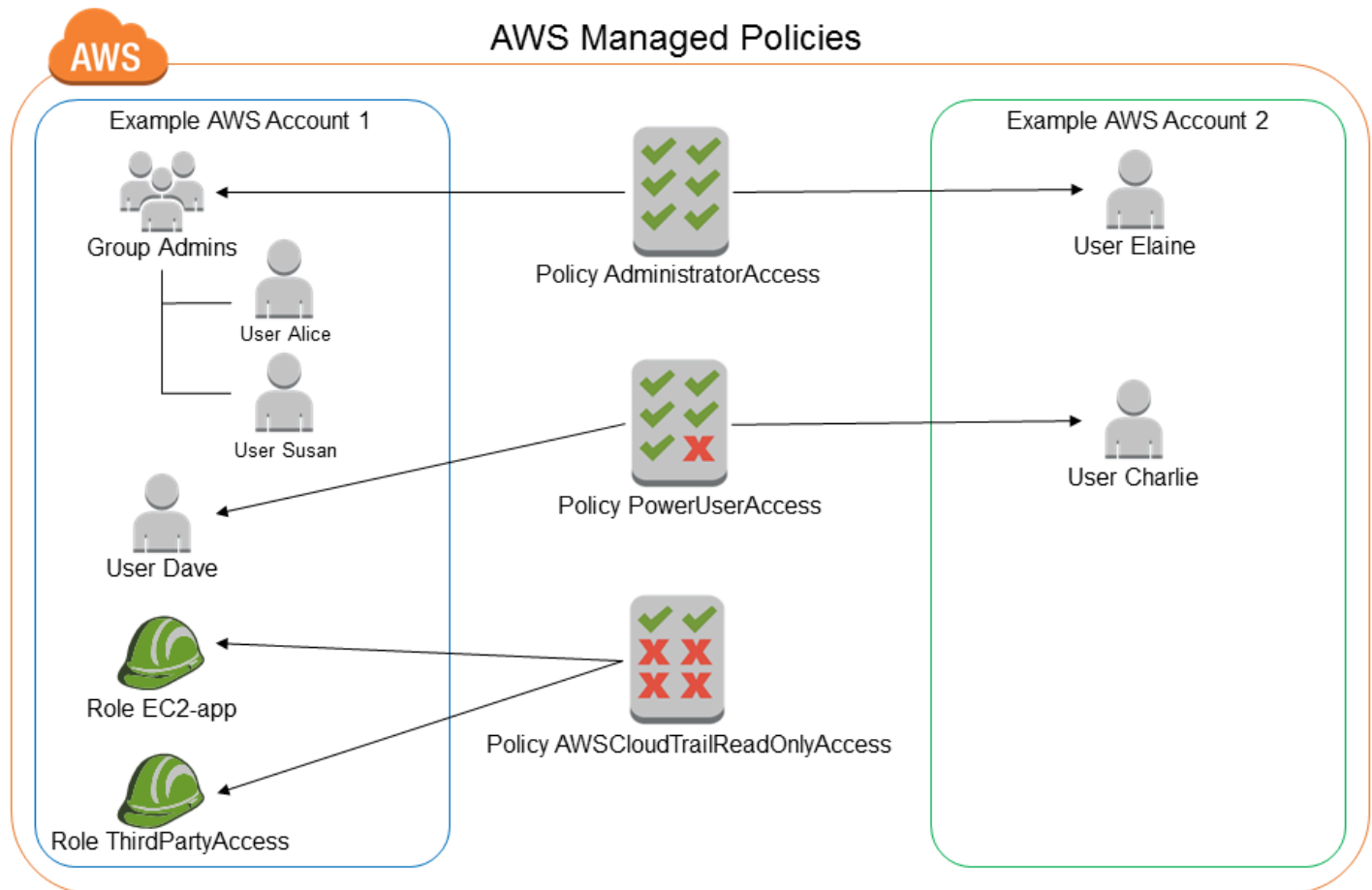
Kebijakan AWS terkelola akses sebagian memberikan tingkat akses tertentu ke AWS layanan tanpa mengizinkan [izin tingkat akses manajemen izin](#).

- [AmazonMobileAnalyticsWriteOnlyAccess](#)
- [AmazonEC2 ReadOnlyAccess](#)

Salah satu kategori kebijakan terkelola AWS yang sangat berguna adalah yang dirancang untuk fungsi pekerjaan. Kebijakan ini selaras erat dengan fungsi pekerjaan yang umum digunakan di industri TI dan memfasilitasi pemberian izin untuk fungsi pekerjaan ini. Satu keuntungan utama dari penggunaan kebijakan fungsi pekerjaan adalah karena itu dipelihara dan diperbarui oleh AWS sebagai layanan baru dan operasi API diperkenalkan. Misalnya, fungsi [AdministratorAccess](#) pekerjaan menyediakan akses penuh dan delegasi izin ke setiap layanan dan sumber daya di AWS. Kami menyarankan Anda menggunakan kebijakan ini hanya untuk administrator akun. Untuk pengguna mahir yang membutuhkan akses penuh ke setiap layanan kecuali akses terbatas ke IAM dan Organizations, gunakan fungsi pekerjaan [PowerUserAccess](#). Untuk daftar dan deskripsi kebijakan fungsi pekerjaan, lihat [AWS kebijakan terkelola untuk fungsi pekerjaan](#).

Diagram berikut menggambarkan kebijakan terkelola AWS. Diagram menunjukkan tiga kebijakan AWS terkelola: `AdministratorAccess`, `PowerUserAccess`, dan `AWSCloudTrailReadOnlyAccess`.

Perhatikan bahwa kebijakan AWS terkelola tunggal dapat dilampirkan ke entitas utama yang berbedaAkun AWS, dan ke entitas utama yang berbeda dalam satuAkun AWS.



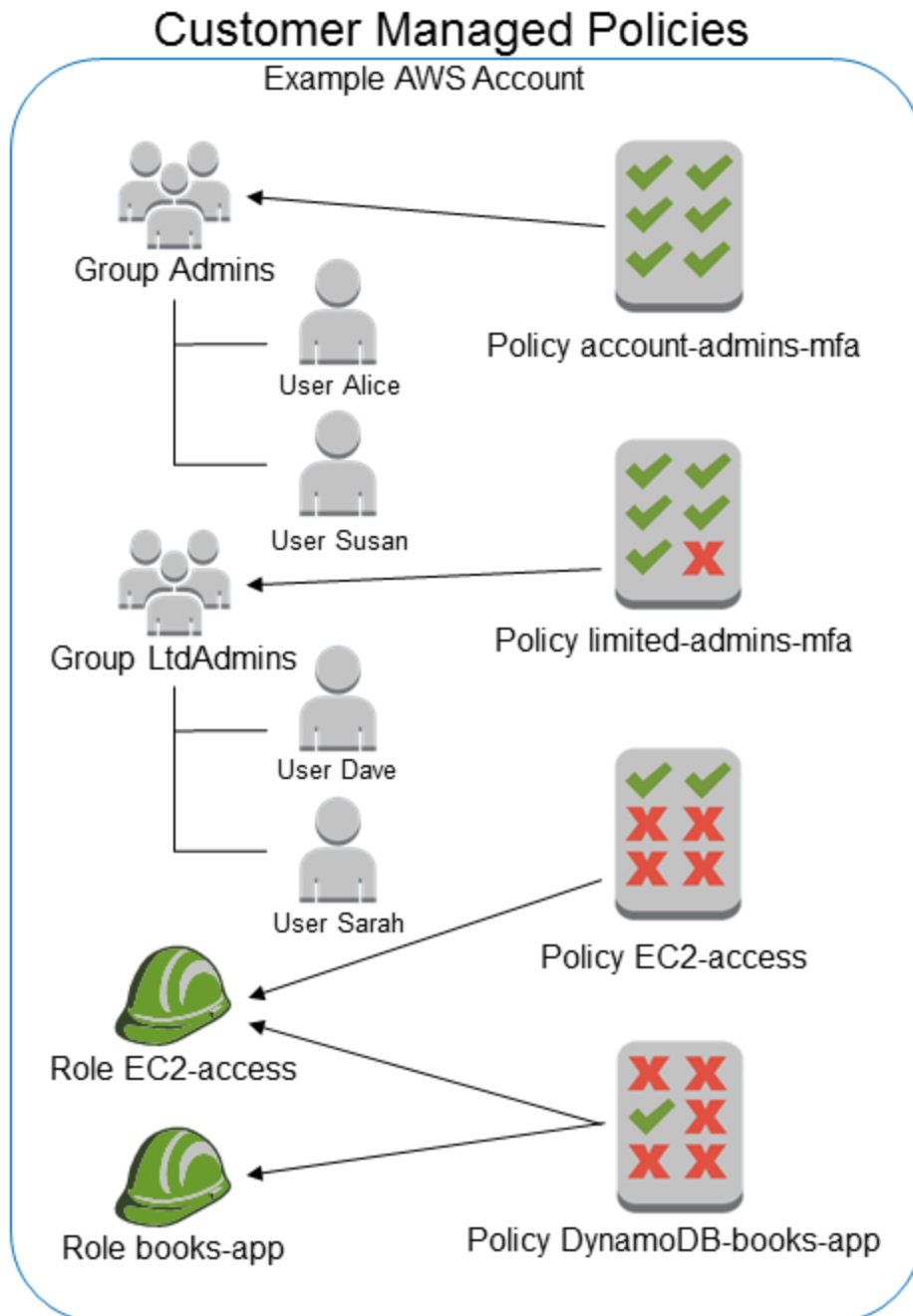
Kebijakan yang dikelola pelanggan

Anda dapat membuat kebijakan mandiri sendiri Akun AWS yang dapat Anda lampirkan ke entitas utama (pengguna, grup, dan peran). Anda membuat kebijakan yang dikelola pelanggan ini untuk kasus penggunaan spesifik Anda, dan Anda dapat mengubah dan memperbaruinya sesering yang Anda suka. Seperti kebijakan AWS terkelola, saat Anda melampirkan kebijakan ke entitas utama, Anda memberi entitas izin yang ditentukan dalam kebijakan tersebut. Saat Anda memperbarui izin dalam kebijakan, perubahan akan 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, kebijakan DynamoDB-Books-App yang sama dilampirkan ke dua peran IAM yang berbeda.

Lihat informasi yang lebih lengkap di [Membuat kebijakan IAM](#)



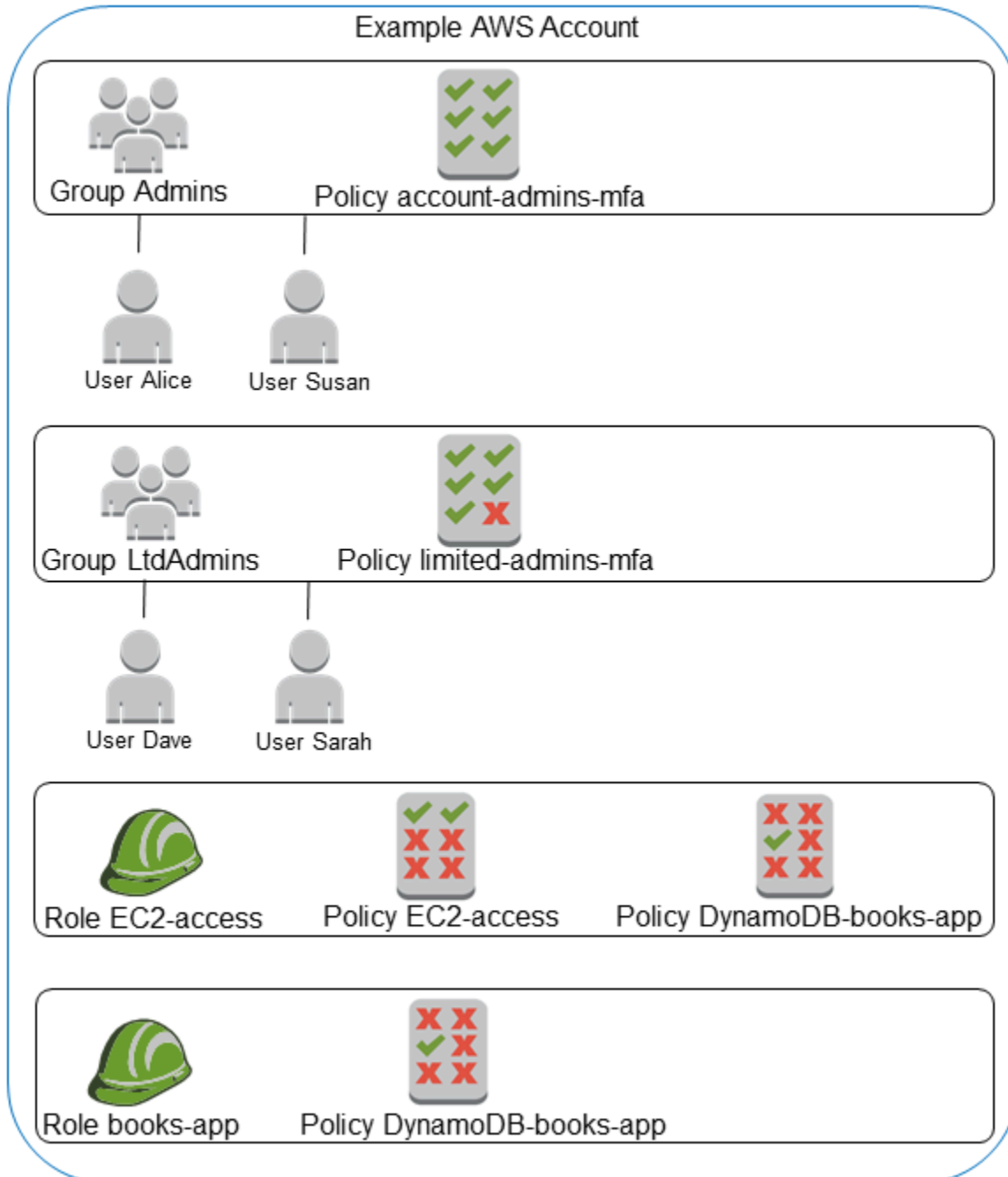
Kebijakan inline

Kebijakan inline adalah kebijakan yang dibuat untuk identitas IAM tunggal (pengguna, grup, atau peran). Kebijakan inline mempertahankan one-to-one hubungan yang ketat antara kebijakan dan 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 berlaku untuk lebih dari satu entitas, lebih baik menggunakan 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 (kebijakan DynamoDB-Books-App), tetapi mereka tidak membagikan kebijakan tunggal. Setiap peran memiliki salinan kebijakan sendiri.

Inline Policies

Example AWS Account



Memilih 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 oleh 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 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. 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, lihat [IAM 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.

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 AWS Management Console untuk menghapus identitas itu, kebijakan yang disematkan dalam identitas akan dihapus juga karena mereka adalah bagian dari entitas utama.

Memulai dengan kebijakan yang dikelola

Sebaiknya gunakan kebijakan yang [memberikan hak akses terkecil](#), atau hanya memberikan izin yang diperlukan untuk melaksanakan tugas. Cara paling aman untuk memberikan hak istimewa paling sedikit adalah dengan menulis kebijakan yang dikelola pelanggan 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 terkelola pelanggan IAM](#) yang hanya menyediakan izin sesuai kebutuhan tim Anda.

Untuk mulai menambahkan izin ke identitas IAM (pengguna, kelompok pengguna, dan peran), Anda dapat menggunakan [AWS Kebijakan yang dikelola](#). AWS kebijakan terkelola tidak memberikan izin hak istimewa paling sedikit. Anda harus mempertimbangkan risiko keamanan pemberian prinsipal Anda izin lebih dari 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 paling sedikit, Anda dapat menjalankan AWS Identity and Access Management Access Analyzer untuk memantau prinsipal dengan kebijakan AWS terkelola. Setelah mempelajari izin yang mereka gunakan, maka 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 cara tim Anda menggunakan AWS. Untuk informasi selengkapnya, lihat [Pembuatan kebijakan IAM Access Analyzer](#).

Kebijakan terkelola AWS dirancang untuk memberikan izin bagi banyak kasus penggunaan umum. Untuk informasi selengkapnya tentang kebijakan AWS terkelola yang dirancang untuk fungsi pekerjaan tertentu, lihat [AWS kebijakan terkelola untuk fungsi pekerjaan](#).

Untuk daftar kebijakan AWS terkelola, lihat [Panduan Referensi Kebijakan AWS Terkelola](#).

Mengonversi kebijakan inline ke 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.

Untuk mengonversi kebijakan inline ke kebijakan terkelola

1. Masuk ke AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.

2. Di panel navigasi, pilih Grup pengguna, Pengguna, atau Peran.
3. Dari daftar, pilih nama grup pengguna, pengguna, atau peran yang memiliki kebijakan yang ingin Anda hapus.
4. Pilih tab Izin.
5. Untuk grup pengguna, pilih nama kebijakan sebaris yang ingin Anda hapus. Untuk pengguna dan peran, pilih Tampilkan **n** lainnya, jika perlu, lalu perluas kebijakan inline 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, lalu 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 pengguna, centang kotak di samping nama kebijakan baru Anda, lalu pilih Add permissions (Tambah kebijakan), lalu pilih Attach policy (Lampirkan kebijakan). Untuk pengguna atau peran, pilih Tambahkan izin. Pada halaman berikutnya, pilih Berikutnya, pilih kotak centang di sebelah nama kebijakan baru Anda, lalu pilih Tambahkan izin.

Anda kembali ke halaman Ringkasan untuk grup pengguna, pengguna, atau peran Anda.

14. centang kotak di sebelah kebijakan inline yang ingin Anda hapus dan pilih Hapus.

Kebijakan terkelola yang tidak digunakan lagi AWS

Untuk menyederhanakan penetapan izin, AWS berikan kebijakan [terkelola —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 ketika 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. Kebijakan terkelola usang muncul dengan ikon peringatan di sebelahnya dalam daftar Kebijakan di konsol IAM.

Kebijakan usang memiliki karakteristik sebagai berikut:

- Itu terus berfungsi untuk semua pengguna, grup, dan peran yang dilampirkan saat ini. Tidak ada yang rusak.
- Itu tidak dapat dilampirkan pada pengguna, grup, atau peran baru. Jika Anda melepasnya dari entri yang ada, Anda tidak dapat melampirkannya kembali.
- Setelah Anda melepasnya dari semua entri saat ini, itu tidak lagi terlihat dan tidak lagi dapat digunakan dengan cara apa pun.

Jika pengguna, grup, atau peran membutuhkan kebijakan tersebut, Anda harus melampirkan kebijakan baru. Saat Anda menerima pemberitahuan bahwa kebijakan telah usang, kami menyarankan agar Anda segera merencanakan untuk melampirkan semua pengguna, grup, dan peran ke kebijakan pengganti dan memisahkannya dari kebijakan yang telah usang tersebut. Terus menggunakan kebijakan yang sudah usang dapat membawa risiko yang hanya dapat dikurangi dengan melakukan pergantian ke kebijakan pengganti.

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. `NotPrincipalElement` 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 `NotPrincipal` elemen, lihat [Elemen kebijakan AWS JSON: NotPrincipal](#).

Anda dapat menggunakan kebijakan AWS terkelola atau kebijakan yang dikelola pelanggan untuk menetapkan batas entitas IAM (pengguna atau peran). Kebijakan tersebut membatasi izin maksimum bagi pengguna atau peran.

Misalnya, asumsikan bahwa nama pengguna IAM `ShirleyRodriguez` harus diizinkan untuk mengelola hanya Amazon S3, Amazon, dan CloudWatch Amazon EC2. Untuk menegakkan aturan ini, Anda dapat menggunakan kebijakan berikut untuk mengatur batas izin untuk pengguna `ShirleyRodriguez`:

```
{
  "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 `ShirleyRodriguez` sebagai semua 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 `ShirleyRodriguez`:

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:CreateUser",
    "Resource": "*"
  }
}
```

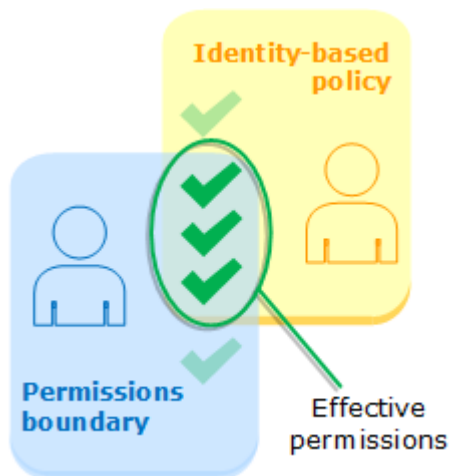
Kebijakan ini memungkinkan pembuatan pengguna di IAM. Jika Anda melampirkan kebijakan izin ini ke pengguna ShirleyRodriguez dan Shirley mencoba membuat sebuah pengguna, operasi gagal. Itu gagal karena batas izin tidak mengizinkan operasi `iam:CreateUser`. Mengingat kedua kebijakan ini, Shirley tidak 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 entitas dapat dipengaruhi oleh kebijakan berbasis identitas, kebijakan berbasis sumber daya, batas izin, Organizations SCP, atau kebijakan sesi. Untuk informasi selengkapnya tentang berbagai jenis kebijakan, lihat [Kebijakan dan Izin di IAM](#).

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 perincian lebih lanjut tentang cara AWS mengevaluasi kebijakan, lihat [Logika evaluasi kebijakan](#).

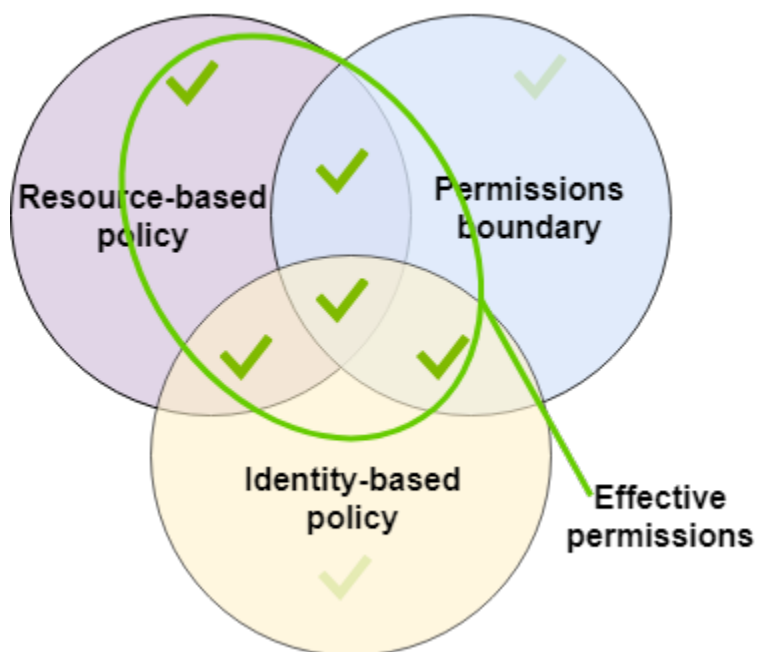
Kebijakan berbasis identitas dengan batasan – Kebijakan berbasis identitas bersifat inline atau kebijakan terkelola yang dilampirkan pada pengguna, grup pengguna, atau peran. Kebijakan berbasis identitas memberikan izin kepada sebuah entitas, dan batasan izin membatasi izin tersebut. Izin yang efektif adalah titik temu dari kedua jenis kebijakan. Penolakan secara tegas dalam salah satu kebijakan ini membatalkan izin.



Kebijakan berbasis sumber daya – Kebijakan berbasis sumber daya mengontrol bagaimana prinsipal tertentu dapat mengakses sumber daya tempat kebijakan tersebut dilampirkan.

Kebijakan berbasis sumber daya untuk pengguna IAM

Dalam akun yang sama, kebijakan berbasis sumber daya yang memberikan izin kepada ARN pengguna IAM (yang bukan sesi pengguna gabungan) tidak dibatasi oleh penolakan implisit dalam kebijakan berbasis identitas atau batas izin.



Kebijakan berbasis sumber daya untuk peran IAM

Peran IAM — Kebijakan berbasis sumber daya yang memberikan izin ke ARN peran IAM dibatasi oleh penolakan implisit dalam batas izin atau kebijakan sesi.

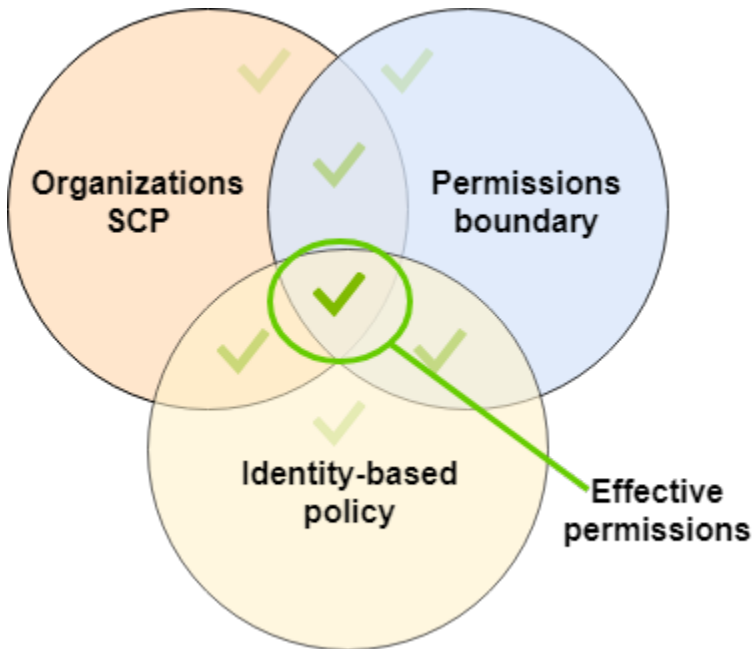
Sesi peran IAM — Dalam akun yang sama, kebijakan berbasis sumber daya yang memberikan izin ke sesi peran IAM ARN memberikan izin langsung ke sesi peran yang diasumsikan. Izin yang diberikan langsung ke sesi tidak dibatasi oleh penolakan implisit dalam kebijakan berbasis identitas, batas izin, atau kebijakan sesi. Ketika Anda mengambil peran dan membuat permintaan, kepala sekolah yang membuat permintaan adalah sesi peran IAM ARN dan bukan ARN dari peran itu sendiri.

Kebijakan berbasis sumber daya untuk sesi pengguna federasi IAM

Sesi pengguna federasi IAM — Sesi pengguna federasi IAM adalah sesi yang dibuat dengan menelepon. [GetFederationToken](#) Ketika pengguna federasi membuat permintaan, kepala sekolah 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.

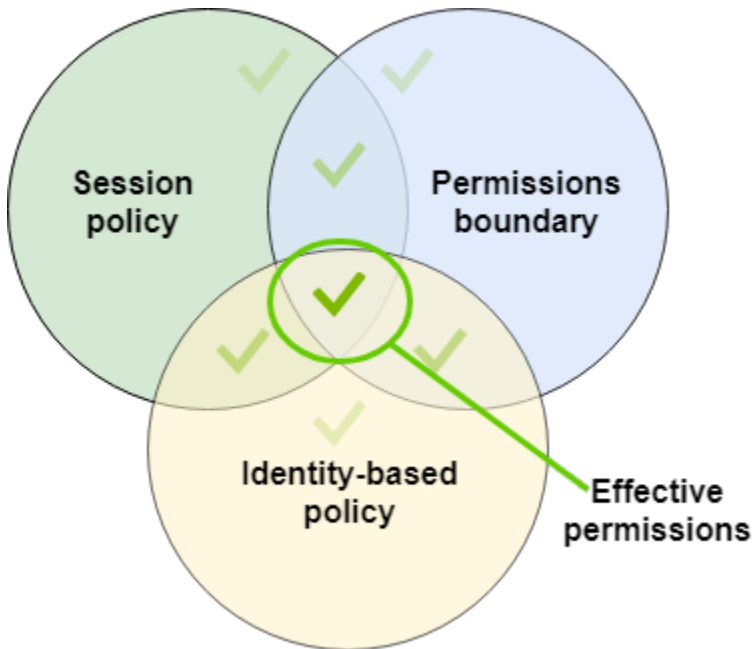
Namun, jika kebijakan berbasis sumber daya memberikan izin kepada ARN pengguna IAM yang berfederasi, maka permintaan yang dibuat oleh pengguna federasi selama sesi dibatasi oleh penolakan implisit dalam batas izin atau kebijakan sesi.

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 sebuah 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 entitas Organizations Anda. Anda harus memiliki izin tambahan untuk melakukan operasi di konsol Organisasi. Untuk mempelajari apakah SCP menolak akses ke permintaan tertentu, atau untuk mengubah izin efektif Anda, hubungi administrator AWS Organizations 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 Akun AWS X-Company. Dia ingin memberikan 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 rinciannya 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 menentukannya sebagai batas izin untuk Zhang. María 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 pengelolaan 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 mengizinkan ini, tambahkan tindakan `*LoginProfile` ke pernyataan `AllowManageOwnPasswordAndAccessKeys`.
- Menolak akses pengguna ke catatan bucket Amazon S3 atau instans Amazon EC2 `i-1234567890abcdef0`.

```
{
  "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. Pernyataan kebijakan `ServiceBoundaries` ini mengizinkan akses penuh ke layanan AWS tertentu. Artinya jika tindakan pengguna baru dalam layanan ini hanya dibatasi oleh kebijakan izin yang terlampir pada pengguna.

2. Pernyataan `AllowIAMConsoleForCredentials` mengizinkan akses untuk mencantumkan semua pengguna IAM. Akses ini diperlukan untuk menavigasi halaman Pengguna di AWS Management Console. Ini juga mengizinkan untuk melihat persyaratan kata sandi untuk akun, yang diperlukan saat mengubah kata sandi Anda sendiri.
3. `AllowManageOwnPasswordAndAccessKeys` Pernyataan ini memungkinkan pengguna untuk mengelola hanya kata sandi konsol mereka sendiri dan kunci akses terprogram. Ini penting jika Zhang atau administrator lain memberikan 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 untuk membuat semua pengguna Perusahaan X, tetapi hanya dengan batas izin `XCompanyBoundaries`. Dia menciptakan kebijakan terkelola pelanggan berikut bernama `DelegatedUserBoundary`. Kebijakan ini menentukan izin maksimum yang dapat dimiliki Zhang.

```
{
  "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 tercantum 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, María 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.

```
{
  "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. `CloudWatchLimited` Pernyataan itu memungkinkan Zhang untuk melakukan lima tindakan. `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 Maria 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 AWS Management Console. 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. Pada halaman Setel izin, Zhang memilih kebijakan `ReadOnlyAccess` izin IAM `FullAccess` dan `AmazonS3` yang memungkinkan Nikhil melakukan pekerjaannya.
3. Zhang melewati bagian Atur batasan izin, melupakan instruksi Maria.
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.

Kebijakan berbasis identitas 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.

Kebijakan berbasis identitas terlampir pada 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 kredensial keamanan IAM-nya sendiri. Kebijakan berbasis identitas dapat [terkelola atau inline](#).

Kebijakan berbasis sumber daya dilampirkan pada sumber daya. Misalnya, Anda dapat melampirkan kebijakan berbasis sumber daya ke bucket Amazon S3, antrian Amazon SQS, titik akhir VPC, kunci enkripsi, serta tabel dan aliran Amazon DynamoDB. AWS Key Management Service 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 mempelajari apakah prinsipal dalam akun di luar zona kepercayaan (organisasi atau akun terpercaya) memiliki akses untuk mengambil peran Anda, lihat [Apa yang dimaksud dengan Penganalisis Akses IAM?](#). Kebijakan berbasis sumber daya hanya bersifat inline, tidak terkelola.

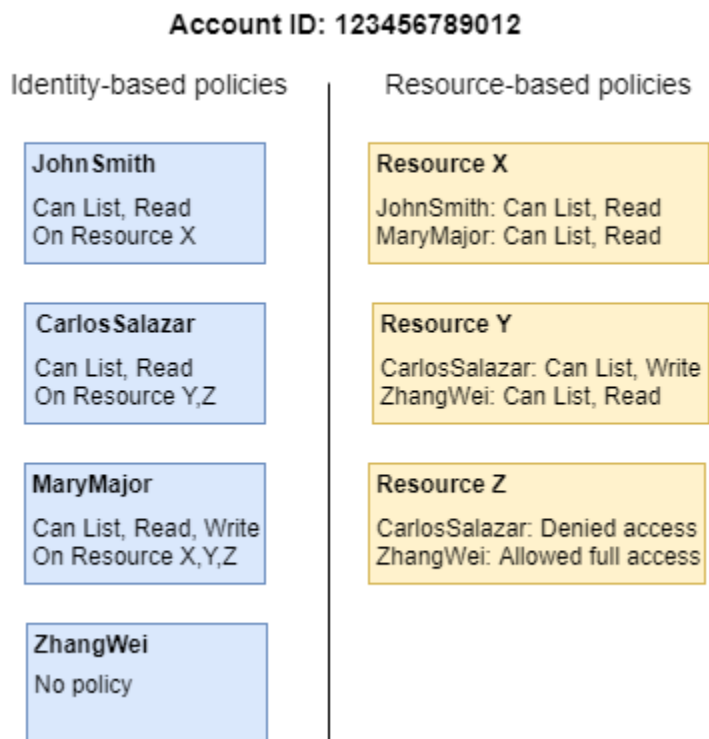
Note

Kebijakan Berbasis sumber daya berbeda dari izin tingkat sumber daya. Anda dapat melampirkan kebijakan berbasis sumber daya secara langsung ke sumber daya, sebagaimana dijelaskan dalam topik ini. Izin tingkat sumber daya mengacu pada kemampuan untuk menggunakan [ARN](#) untuk menentukan sumber daya individu dalam kebijakan. Kebijakan berbasis sumber daya hanya didukung oleh beberapa layanan. AWS 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 [Mengevaluasi kebijakan dalam satu akun](#).

Untuk mempelajari cara interaksi kebijakan di seluruh akun, lihat [Logika evaluasi kebijakan lintas akun](#).

Untuk lebih memahami konsep ini, lihat gambar berikut. Administrator dari akun 123456789012 terlampir kebijakan berbasis identitas kepada pengguna JohnSmith, CarlosSalazar, dan MaryMajor. Beberapa tindakan dalam kebijakan ini dapat dilakukan pada sumber daya tertentu. Misalnya, pengguna JohnSmith 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, Resource Y, dan Resource Z. Kebijakan berbasis sumber daya memungkinkan Anda menentukan siapa yang dapat mengakses sumber daya tersebut. Misalnya, kebijakan berbasis sumber daya di Resource X mengizinkan akses pengguna JohnSmith dan MaryMajor daftar dan baca ke sumber daya.



Contoh akun 123456789012 mengizinkan pengguna berikut untuk melakukan tindakan yang tercantum:

- **JohnSmith**— John dapat melakukan daftar dan membaca tindakan di `Resource X`. Dia diberikan izin ini oleh kebijakan berbasis identitas pada penggunaannya dan kebijakan berbasis sumber daya pada `Resource X`.
- **CarlosSalazar**— 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](#).
- **MaryMajor**— 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.
- **ZhangWei**— 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.

Kebijakan berbasis identitas dan kebijakan berbasis sumber daya keduanya adalah kebijakan izin dan dievaluasi bersama. Untuk permintaan yang hanya berlaku kebijakan izin, AWS pertamanya memeriksa 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.

Mengontrol akses ke sumber daya AWS 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 di AWS, Anda harus memahami bagaimana AWS memberikan akses. AWS terdiri atas kumpulan sumber daya. Pengguna IAM adalah sebuah sumber daya. Bucket Amazon S3 adalah sebuah sumber daya. Saat Anda menggunakan API AWS, AWS CLI, atau AWS Management Console untuk melakukan operasi (seperti membuat pengguna), Anda mengirim 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.

Kemudian AWS memeriksa bahwa Anda (penanggung jawab) diautentikasi (masuk) dan diberi otorisasi (memiliki izin) untuk melakukan tindakan tertentu pada sumber daya yang ditentukan. Selama otorisasi, AWS memeriksa semua kebijakan yang berlaku pada konteks permintaan Anda. Sebagian besar kebijakan disimpan di AWS sebagai [Dokumen JSON](#) dan menentukan izin untuk entitas penanggung jawab. Untuk informasi selengkapnya tentang jenis dan penggunaan kebijakan, lihat [Kebijakan dan Izin di IAM](#).

AWS mengizinkan 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, lihat [Logika 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 pengguna, 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.
- [AWSSumber Daya](#) — Kontrol siapa yang memiliki akses ke sumber daya menggunakan kebijakan berbasis identitas atau kebijakan berbasis sumber daya.
- [AWSAkun](#) — Kontrol apakah permintaan hanya diperbolehkan untuk anggota akun tertentu.

Kebijakan membiarkan Anda menentukan siapa yang memiliki akses ke sumber daya AWS, dan tindakan apa yang dapat mereka lakukan di sumber daya itu. 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 mendeskripsikan instans Amazon EC2. Akun AWS

Untuk informasi tentang cara memberikan izin dasar bagi pengguna, grup pengguna, 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 *DOC-EXAMPLE-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. *DOC-EXAMPLE-BUCKET1* Karena batas izin tidak mengizinkan tindakan IAM apa pun, hal itu mencegah Zhang menghapus batas miliknya (atau siapa pun).

```
{
  "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:::DOC-EXAMPLE-BUCKET1",
        "arn:aws:s3:::DOC-EXAMPLE-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 [Membuat kebijakan IAM](#) .
- 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: Mengizinkan akses penuh EC2 di 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 dalam 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 AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Di panel navigasi di sebelah kiri, pilih Kebijakan.

Jika ini pertama kalinya Anda memilih Kebijakan, akan muncul laman 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 `katagroup`. 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 Tambahkan ARN. Untuk Sumber Daya di, pilih opsi Akun apa pun. Pilih kotak centang Nama grup apa pun dengan jalur lalu ketik nama grup pengguna **AllUsers**. Kemudian pilih Tambahkan ARN.
 - 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 `katagroup`. 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 ARN. Untuk Sumber Daya di, pilih opsi Akun apa pun. Untuk Setiap nama grup dengan jalur, ketik nama grup pengguna **AllUsers**. Kemudian pilih Tambahkan ARN.
 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 dalam 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 bagaimana pengguna Anda dapat menerapkan kebijakan yang dikelola AWS. 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 [IAM UserChangePassword](#) dan kebijakan [PowerUserAccess](#) 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 memberikan dan melepaskan kebijakan ke dan dari entitas prinsipal (grup pengguna, 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 [Membuat kebijakan IAM](#).
- 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 ke 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 AndaAkun 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 diAkun 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

```
{
  "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

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "iam:DeletePolicyVersion",
      "iam:SetDefaultPolicyVersion"
    ],
    "Resource": "arn:aws:iam::account-id: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 kebijakan terkelola dan/atau entitas prinsipal 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 mengizinkan pengguna untuk memberikan kebijakan terkelola hanya ke grup pengguna dan peran yang mencakup jalur /TEAM-A/. Grup pengguna dan ARN peran ditentukan dalam elemen `Resource` dari kebijakan. (Dalam contoh ini, ARN menyertakan jalur dan karakter wildcard, sehingga cocok dengan semua grup pengguna dan peran 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”](#).

Example Contoh kebijakan yang mengizinkan memberikan kebijakan terkelola hanya untuk grup pengguna atau peran tertentu

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "iam:AttachGroupPolicy",
      "iam:AttachRolePolicy"
    ]
  }
}
```

```
    ],
    "Resource": [
      "arn:aws:iam::account-id:group/TEAM-A/*",
      "arn:aws:iam::account-id: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 mengizinkan pengguna untuk memberikan hanya kebijakan terkelola yang mencakup jalur `/TEAM-A/` hanya ke grup pengguna dan peran 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”](#).

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "iam:AttachGroupPolicy",
      "iam:AttachRolePolicy"
    ],
    "Resource": [
      "arn:aws:iam::account-id:group/TEAM-A/*",
      "arn:aws:iam::account-id:role/TEAM-A/*"
    ],
    "Condition": {"ArnLike":
      {"iam:PolicyARN": "arn:aws:iam::account-id: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 dari kebijakan yang dikelola AWS dalam elemen kebijakan Condition. ARN dari kebijakan terkelola AWS menggunakan alias khusus aws dalam ARN kebijakan sebagai ganti sebuah 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 [Kebijakan berbasis identitas 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 [Membuat kebijakan IAM](#).
- 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 kredensial Pengguna root akun AWS, Anda memiliki izin untuk melakukan tindakan apa pun pada sumber daya yang menjadi milik akun tersebut. 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 [Menyediakan akses ke pengguna IAM di pengguna lain Akun AWS yang Anda miliki](#).

Note

Beberapa layanan mendukung kebijakan berbasis sumber daya sebagaimana dijelaskan dalam [Kebijakan berbasis identitas 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 antrian) 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, lihat [Menandai sumber daya IAM](#).

Note

Untuk detail tentang sensitivitas huruf besar untuk kunci tag dan nilai kunci tag, lihat [Case 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.

Untuk 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 kondisi `aws:ResourceTag/key-name` untuk menentukan pasangan nilai kunci tag mana yang harus dilampirkan ke sumber daya. Untuk informasi selengkapnya, lihat [Mengontrol akses ke sumber daya AWS](#).
- [Permintaan](#) – Mengontrol tag yang dapat diberikan dalam permintaan IAM. Untuk melakukan ini, gunakan kunci kondisi **nama kunci** `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 permintaan AWS](#).
- [Prinsipal](#) – Mengontrol apa yang boleh dilakukan oleh orang yang membuat permintaan (principal) berdasarkan tag yang dilampirkan ke pengguna atau peran IAM orang tersebut. Untuk melakukan ini, gunakan kunci kondisi **nama kunci** `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. Untuk detailnya, lihat [Membuat kebijakan IAM](#).

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 *teks placeholder yang dicetak miring* dalam kebijakan contoh dengan informasi Anda sendiri. Lalu, ikuti petunjuk di [buat kebijakan](#) atau [ubah kebijakan](#).

```
{
  "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 *teks placeholder yang dicetak miring* dalam kebijakan contoh dengan informasi Anda sendiri. Lalu, ikuti petunjuk di [buat kebijakan](#) atau [ubah kebijakan](#).

```
{
  "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 [Menandai sumber daya IAM](#). 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: Menentukan izin untuk mengakses sumber daya AWS berdasarkan tanda](#). 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 AWS Management Console 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 memeriksa 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 IAM](#).

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 kondisi ***nama kunci 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 kondisi `aws:RequestTag/key-name` 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. Untuk detailnya, lihat [Membuat kebijakan IAM](#).

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 sumber daya AWS

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 [Memberi izin kepada pengguna untuk meneruskan peran ke layanan AWS](#).

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.

```
{
  "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-roe` mencoba memulai instans Amazon EC2, instans harus diberi tag `Owner=richard-roe` atau `owner=richard-roe`. 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: Condition](#).

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang menggunakan tag `team` utama di ARN sumber daya. Kebijakan memberikan izin untuk menghapus antrian Layanan Antrian Sederhana Amazon, tetapi hanya jika nama antrian dimulai dengan nama tim yang diikuti. `-queue` Misalnya, `qa-queue` jika `qa` adalah nama tim untuk tag `team` utama.

```

{
    "Version": "2012-10-17",
    "Statement": {
        "Sid": "AllQueueActions",
        "Effect": "Allow",
        "Action": "sqs:DeleteQueue",
        "Resource": "arn:aws:sqs:us-east-2:${aws:PrincipalTag/team}-queue"
    }
}

```

Mengontrol akses selama permintaan AWS

Anda dapat menggunakan kondisi dalam kebijakan IAM untuk mengontrol pasangan nilai kunci tag apa yang dapat diteruskan dalam permintaan yang menerapkan tag 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`.

```
{
  "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 Pengguna Organizations.

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 Anda menggunakan IAM. Untuk melakukannya, Anda dapat melampirkan kebijakan sumber daya secara 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 (prinsipal 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, lihat [Kebijakan berbasis identitas dan kebijakan berbasis sumber daya](#).

Note

Peran IAM dan kebijakan berbasis sumber daya mendelegasikan akses ke seluruh akun hanya dalam satu partisi tunggal. Misalnya, Anda memiliki akun di AS Barat (California N.) dalam aws partisi standar. Anda juga memiliki akun di China dalam aws-cn partisi. Anda tidak dapat menggunakan kebijakan berbasis sumber daya di akun Anda di Indonesia untuk mengizinkan akses bagi pengguna di akun standar Anda. AWS

Akses lintas akun 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 akun lain untuk mengambil peran tersebut. AWS Sederhananya, Anda dapat membuat peran dalam satu AWS akun yang mendelegasikan izin tertentu ke akun lain AWS.

Untuk informasi tentang melampirkan kebijakan ke identitas IAM, lihat [Mengelola kebijakan IAM](#).

Note

Saat prinsipal beralih ke peran untuk sementara menggunakan izin peran, mereka menyerahkan izin aslinya dan mengambil izin yang ditetapkan untuk peran yang mereka asumsikan.

Mari kita lihat keseluruhan proses yang berlaku untuk perangkat lunak APN Partner 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 perannya adalah `APNPartner`.

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

```

        "Action": "s3:*",
        "Resource": [
            "arn:aws:s3:::bucket-name"
        ]
    }
]
}

```

2. Kemudian, pelanggan menentukan bahwa peran tersebut dapat diasumsikan oleh AWS akun mitra dengan memberikan Akun AWS ID Mitra APN dalam [kebijakan kepercayaan](#) untuk peran tersebut `APNPartner`.

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::APN-account-ID:role/APN-user-name"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}

```

3. Pelanggan memberikan Amazon Resource Name (ARN) peran tersebut kepada mitra APN. ARN adalah nama peran yang sepenuhnya memenuhi syarat.

```
arn:aws:iam::APN-ACCOUNT-ID:role/APNPartner
```

Note

Sebaiknya gunakan ID eksternal dalam situasi multi-penyewa. Untuk detailnya, lihat [Cara menggunakan ID eksternal saat memberikan akses ke sumber daya AWS Anda kepada pihak ketiga](#).

4. Ketika perangkat lunak Mitra APN perlu mengakses akun pelanggan, perangkat lunak memanggil `AssumeRole` API AWS Security Token Service dengan ARN peran dalam akun pelanggan. STS mengembalikan AWS kredensi sementara yang memungkinkan perangkat lunak untuk melakukan tugasnya.

Untuk contoh lain pemberian akses lintas akun menggunakan peran, lihat [Menyediakan akses ke pengguna IAM di pengguna lain](#) Akun AWS yang Anda miliki. Anda juga dapat mengikuti [Tutorial IAM: Delegasikan akses AWS akun yang menggunakan peran IAM](#).

Akses lintas akun menggunakan kebijakan berbasis sumber daya

Saat akun mengakses sumber daya melalui akun lain menggunakan kebijakan berbasis sumber daya, prinsipal masih bekerja di akun tepercaya dan tidak harus menyerahkan izinnya untuk menerima izin peran. Dengan kata lain, kepala sekolah terus memiliki akses ke sumber daya di akun tepercaya sambil memiliki akses ke sumber daya dalam akun kepercayaan. 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 mencakup akun, pengguna IAM, pengguna federasi, peran IAM, sesi peran yang diasumsikan, atau layanan. AWS Untuk informasi selengkapnya, lihat [Menentukan kepala sekolah](#).

Untuk mengetahui apakah kepala sekolah di akun di luar zona kepercayaan Anda (organisasi atau akun tepercaya) memiliki akses untuk mengambil peran Anda, lihat [Mengidentifikasi sumber daya yang dibagikan dengan entitas eksternal](#).

Daftar berikut mencakup beberapa layanan AWS yang mendukung kebijakan berbasis sumber daya. Untuk daftar lengkap jumlah layanan AWS yang terus berkembang dan mendukung pelampiran kebijakan izin ke sumber daya alih-alih prinsipal, lihat [AWS layanan yang bekerja dengan IAM](#) dan cari layanan yang memiliki Ya dalam kolom Berbasis Sumber Daya.

- Bucket Amazon S3 — Kebijakan ini dilampirkan ke bucket, tetapi kebijakan mengontrol akses ke bucket dan objek di dalamnya. Untuk informasi selengkapnya, lihat [Kontrol Akses](#) 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 penelusuran dalam Panduan](#) Pengguna Amazon Simple Storage Service.
- Topik Amazon Simple Notification Service (Amazon SNS) — Untuk informasi selengkapnya, kunjungi [Contoh kasus untuk kontrol akses Amazon SNS](#) — Untuk informasi selengkapnya, kunjungi [Contoh kasus untuk kontrol akses Amazon SNS](#) di Panduan Pengembang Amazon Simple Notification Service.
- Antrian Amazon Simple Queue Service (Amazon SQS) — Untuk informasi selengkapnya, kunjungi [Lampiran: Bahasa Kebijakan Akses dalam Panduan Pengembang](#) Amazon Simple Queue Service.

Mendelegasikan AWS izin dalam kebijakan berbasis sumber daya

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 Allow kebijakan identitas dan kebijakan berbasis sumber daya.

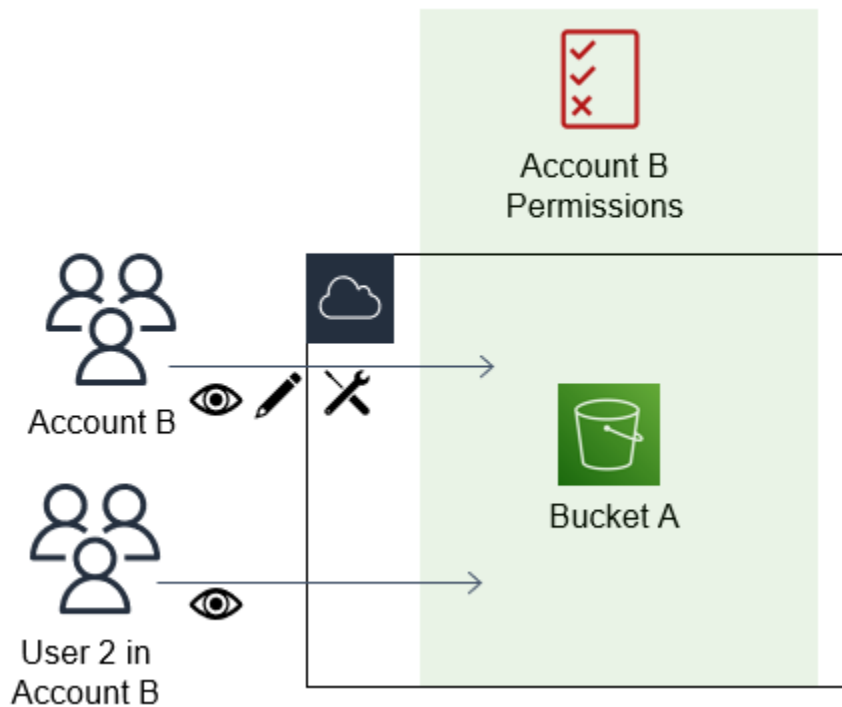
Asumsikan bahwa kebijakan berbasis sumber daya memberi semua prinsipal di akun Anda akses administratif penuh ke sumber daya. Kemudian Anda dapat mendelegasikan akses penuh, akses hanya baca, atau akses parsial lainnya ke prinsipal di akun AWS Anda. Alternatifnya, jika kebijakan berbasis sumber daya 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 [Menentukan apakah permintaan diizinkan atau ditolak dalam akun](#).

Note

Peran IAM dan kebijakan berbasis sumber daya mendelegasikan akses ke seluruh akun hanya dalam satu partisi tunggal. 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 untuk BucketA yang memungkinkan semua kepala sekolah di accountB akses penuh ke objek dalam bucket Anda. Mereka dapat membuat, membaca, atau menghapus objek apa pun dalam bucket tersebut.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PrincipalAccess",
      "Effect": "Allow",
      "Principal": {"AWS": "arn:aws:iam::AccountB:root"},
      "Action": "s3:*",
      "Resource": "arn:aws:s3:::BucketA/*"
    }
  ]
}
```

AccountA memberikan akses penuh accountTB ke BucketA dengan menamai accountTB sebagai prinsipal dalam kebijakan berbasis sumber daya. Akibatnya, accountTB berwenang untuk melakukan tindakan apa pun pada BucketA, dan administrator accountB dapat mendelegasikan akses ke penggunanya dalam accountTB.

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 tersebut memungkinkan akses hanya-baca pengguna ke objek di BuckEta. Itu berarti bahwa User2 dapat melihat objek, tetapi tidak membuat, mengedit, atau menghapusnya.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect" : "Allow",
      "Action" : [
        "s3:Get*",
        "s3:List*" ],
      "Resource" : "arn:aws:s3:::BucketA/*"
    }
  ]
}
```

Tingkat akses maksimum yang dapat didelegasikan oleh accountB adalah tingkat akses yang diberikan ke akun. Dalam kasus ini, kebijakan berbasis sumber daya memberikan akses penuh ke accountB, tetapi User2 hanya diberikan akses read-only.

Administrator AccountB tidak memberikan akses ke User1. Secara default, pengguna tidak memiliki izin apa pun kecuali yang secara eksplisit diberikan, 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 User2 yang memungkinkan akses penuh ke semua sumber daya di semua akun, User2 akan secara otomatis memiliki akses ke sumber daya apa pun yang dapat diakses oleh 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, lihat [Skenario umum untuk peran: Pengguna, aplikasi, dan layanan](#).

Important

Berikan akses hanya kepada entitas yang Anda percayai, dan berikan tingkat akses minimum yang diperlukan. Setiap kali entitas tepercaya adalah AWS akun lain, setiap prinsipal IAM dapat diberikan akses ke sumber daya Anda. Akun AWS tepercaya dapat mendelegasikan akses hanya jika sudah diberi akses; akun tidak dapat mendelegasikan lebih banyak akses dari yang didelegasikan kepadanya.

Untuk informasi tentang izin, kebijakan, dan bahasa kebijakan izin yang Anda gunakan untuk menyusun kebijakan, lihat [Manajemen akses untuk sumber daya AWS](#).

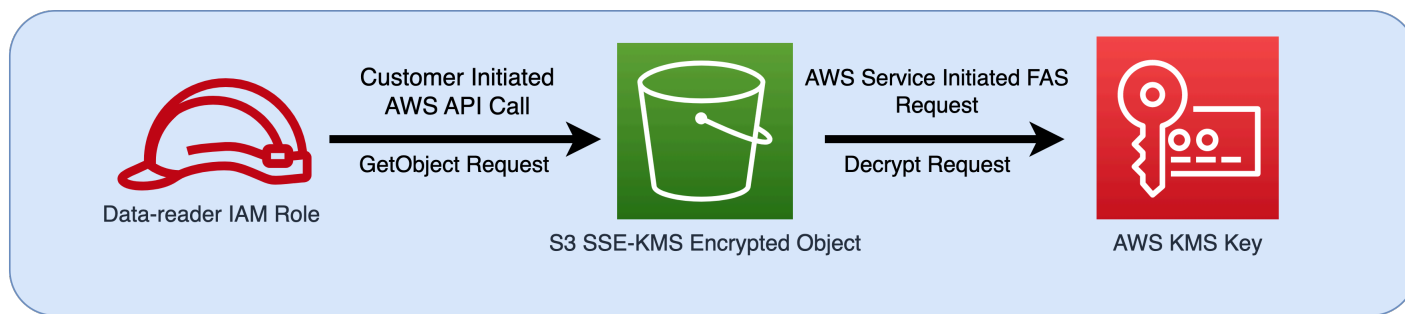
Teruskan sesi akses

Forward Access Sessions (FAS) adalah teknologi IAM yang digunakan oleh AWS layanan untuk meneruskan identitas, izin, dan atribut sesi Anda saat AWS layanan membuat permintaan atas nama Anda. FAS menggunakan izin identitas yang memanggil AWS layanan, dikombinasikan dengan identitas AWS layanan untuk membuat permintaan ke layanan hilir. Permintaan FAS hanya dibuat untuk AWS layanan atas nama kepala 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 kepala IAM memeriksa izin dari 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 [ketika](#) SSE-KMS digunakan untuk mengenkripsinya. Saat mengunduh objek terenkripsi SSE-KMS, peran bernama pembaca data memanggil GetObject objek terhadap Amazon S3, dan tidak memanggil secara langsung. AWS KMS 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 untuk objek Amazon S3 dan kunci KMS. AWS

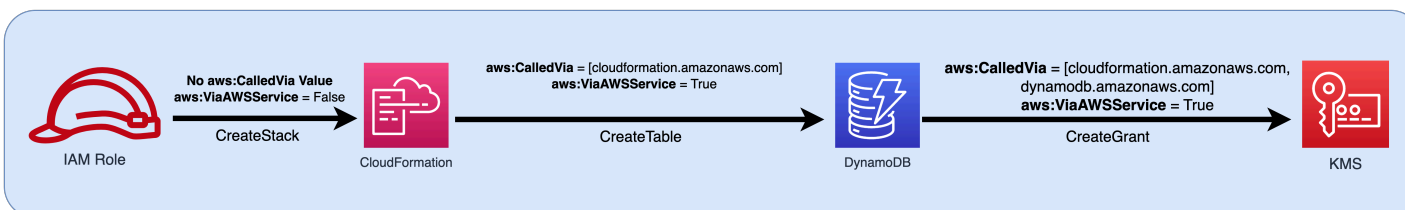


Note

Permintaan FAS tambahan dapat dilakukan 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 dibuat, [aws: CalledVia](#), [aws: CalledViaFirst](#), dan kunci [aws: CalledViaLast](#) kondisi diisi dengan prinsip layanan layanan yang memulai panggilan FAS. Nilai kunci [AWS: melalui AWSService](#) kondisi diatur ke true setiap kali permintaan FAS dibuat. Dalam diagram berikut, permintaan untuk CloudFormation secara langsung tidak memiliki `aws:CalledVia` atau kunci `aws:ViaAWSService` kondisi yang ditetapkan. Kapan CloudFormation dan DynamoDB membuat permintaan FAS hilir atas nama peran, nilai untuk kunci kondisi ini diisi.



Untuk mengizinkan permintaan FAS dibuat jika tidak akan ditolak oleh pernyataan kebijakan Deny dengan kunci kondisi yang menguji alamat IP Sumber atau VPC Sumber, Anda harus menggunakan kunci kondisi untuk memberikan pengecualian untuk permintaan FAS dalam kebijakan Deny 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 `aws: SourceVpce` untuk `aws: SourceVpc`, `aws: VpcSourceIp` dan dari permintaan awal tidak digunakan dalam permintaan FAS. Saat menulis kebijakan menggunakan `aws: VPCSourceIP` atau `aws: SourceVPCE` memberikan akses secara kondisional, Anda juga harus menggunakan `aws: ViaAWSService` atau `aws: CalledVia` mengizinkan 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 GetObject S3 dan Athena hanya diizinkan jika permintaan tersebut berasal dari titik akhir VPC yang dilampirkan *ke* `example_vpc`, atau jika permintaan tersebut adalah permintaan FAS yang dibuat oleh Athena.

```
{
  "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": [
            "example_vpc"
          ]
        }
      }
    }
  ],
}
```

```
{
  "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 contoh 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 untuk mengelola kata sandi mereka sendiri di halaman Kredensial 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 instans Amazon EC2 berdasarkan tag ([Lihat kebijakan ini.](#))
- Mengizinkan peluncuran instans Amazon EC2 dalam subnet tertentu, secara terprogram dan di konsol ([Lihat kebijakan ini.](#))
- Mengizinkan pengelolaan grup keamanan Amazon EC2 terkait dengan VPC tertentu, secara terprogram dan di konsol ([Lihat kebijakan ini.](#))
- Mengizinkan memulai atau menghentikan instans Amazon EC2 yang telah ditandai pengguna, secara terprogram, dan di konsol ([Lihat kebijakan ini.](#))
- Mengizinkan memulai atau menghentikan instans Amazon EC2 berbasis sumber daya dan tag prinsipal, secara terprogram, dan di konsol ([Lihat kebijakan ini.](#))
- Mengizinkan memulai atau menghentikan instans Amazon EC2 ketika sumber daya dan tag prinsipal cocok ([Lihat kebijakan ini.](#))
- Mengizinkan akses penuh Amazon EC2 dalam Wilayah tertentu, secara terprogram dan di konsol. ([Lihat kebijakan ini.](#))
- Mengizinkan memulai atau menghentikan instans Amazon EC2 tertentu dan mengubah grup keamanan tertentu, secara terprogram dan di konsol ([Lihat kebijakan ini.](#))
- Menolak akses ke operasi Amazon EC2 tertentu tanpa MFA ([Lihat kebijakan ini.](#))
- Membatasi akhir instans Amazon EC2 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.](#))
- Mengizinkan pengguna gabungan untuk mengakses direktori home milik mereka 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 bagaimana Anda dapat membuat kebijakan berbasis identitas yang mengizinkan 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 memberi izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari AWS API atau AWS CLI. Untuk menggunakan kebijakan ini, ganti *teks placeholder cetak miring* dalam contoh kebijakan 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](#).

```
{
  "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: Mengizinkan untuk mengaktifkan dan menonaktifkan AWS wilayah AWS

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan administrator untuk mengaktifkan dan menonaktifkan Wilayah Asia Pasifik (Hong Kong) (ap-east-1). Kebijakan ini mendefinisikan izin untuk akses terprogram dan konsol. Pengaturan ini muncul di halaman Pengaturan akun di AWS Management Console. Halaman ini mencakup informasi tingkat akun yang sensitif yang harus dilihat dan dikelola hanya oleh administrator akun. Untuk menggunakan kebijakan ini, ganti *teks placeholder yang dicetak miring* dalam contoh kebijakan ini untuk menggunakan kebijakan ini untuk mengaktifkan dan menggunakan kebijakan ini untuk mengaktifkan dan menggunakan kebijakan ini untuk mengaktifkan dan menggunakan kebijakan ini untuk mengaktifkan dan menggunakan kebijakan ini untuk mengaktifkan dan mengganti 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.

```
{
  "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 AWS Management Console 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 AWS Management Console 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 mengotentikasi dengan MFA. Untuk informasi selengkapnya, lihat [Mengonfigurasi akses API yang dilindungi 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 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 [aws: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`.

```
{
  "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 logisAND. 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 memberi izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari AWS API atau AWS CLI. Untuk menggunakan kebijakan ini, ganti *teks placeholder dicetak miring* pada contoh kebijakan 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](#).

```
{
  "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. AWS Management Console 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 mengizinkan 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`.

```
{
  "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. AWS Management Console 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 [GetSessionToken](#) operasi 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 `AllowManageOwnUserMFADevice` 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](#).

```
{
```

```
"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 AWS Management Console kata sandi mereka sendiri di halaman kredensial Keamanan. AWS Management Console 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 mengizinkan 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 `ChangeOwnPasswords` pernyataan.

```
{
```

```
"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 AWS Management Console 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 mengizinkan 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.

```
{
  "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 ke AWS berdasarkan Wilayah yang diminta

Contoh ini menunjukkan bagaimana Anda dapat 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 mendefinisikan izin untuk akses terprogram dan konsol.

Untuk menggunakan kebijakan ini, ganti *teks placeholder cetak miring miring* dalam contoh kebijakan dengan informasi pribadi. 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 di CloudFront, IAM, Route 53, dan AWS Support layanan tidak boleh ditolak karena merupakan layanan AWS global populer dengan titik akhir tunggal yang secara fisik berlokasi 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, seperti `budgets`, `globalaccelerator`, `importexport`, `organizations`, atau `waf`. Beberapa layanan global lainnya, seperti AWS Chatbot dan AWS Device Farm, adalah layanan global dengan titik akhir yang secara fisik terletak di wilayah us-west-2. Untuk mempelajari tentang semua layanan yang memiliki titik akhir global tunggal, lihat [Wilayah 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.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyAllOutsideRequestedRegions",
      "Effect": "Deny",
      "NotAction": [
        "cloudfront:*",
        "iam:*",
        "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, lihat [AWS kunci konteks kondisi global](#).

```
{  
  "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 *teks placeholder yang dicetak miring* dalam kebijakan contoh 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 menjalankan API. AWS Data Exchange 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 `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 tegas memblokir akses ke semua sumber daya yang tercantum dalam pernyataan yang bukan milik akun yang tercantum. 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.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyAllAwsResourcesOutsideAccountExceptAmazonS3",
      "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 yang tidak dibuat pengguna

Contoh ini menunjukkan bagaimana Anda mungkin membuat kebijakan berbasis identitas yang menolak akses ke saluran yang tidak dibuat pengguna. Jika nilai kolom `PipelineCreator` cocok dengan nama pengguna IAM, maka tindakan yang ditentukan tidak ditolak. Kebijakan ini memberi izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari AWS API atau AWS CLI.

Important

Kebijakan ini tidak mengizinkan tindakan apa pun. Gunakan kebijakan ini bersama dengan kebijakan lain yang mengizinkan tindakan tertentu.

```

{
  "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: Mengizinkan akses ke tabel tertentu

Contoh ini menunjukkan bagaimana Anda mungkin membuat kebijakan berbasis identitas yang mengizinkan akses penuh ke tabel `MyTable` DynamoDB. Kebijakan ini memberi izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari AWS API atau AWS CLI. Untuk menggunakan kebijakan ini, ganti *teks placeholder cetak miring* dalam contoh kebijakan dengan informasi Anda sendiri. Lalu, ikuti petunjuk di [buat kebijakan](#) atau [ubah kebijakan](#).

Important

Kebijakan ini mengizinkan semua tindakan yang dapat dilakukan di tabel DynamoDB.

Untuk meninjau tindakan ini, lihat [Izin API DynamoDB: Referensi Tindakan, dan Kondisi](#) dalam Panduan Developer Amazon DynamoDB. Anda dapat memberikan izin yang sama

dengan mencantumkan setiap tindakan individu. Namun, jika Anda menggunakan wildcard (*) dalam Action elemen tersebut, seperti "dynamodb:List*", maka Anda tidak perlu memperbarui kebijakan jika DynamoDB menambahkan Daftar tindakan baru.

Kebijakan ini mengizinkan tindakan hanya di tabel DynamoDB yang ada dengan nama yang ditentukan. Untuk memungkinkan penggunaRead mengakses semua yang ada di DynamoDB, Anda juga dapat melampirkan kebijakan yangReadOnlyAccessAWS dikelola [AmazonDynamoDB](#).

```
{
  "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: Izinkan akses ke atribut tertentu

Contoh ini menunjukkan bagaimana Anda mungkin membuat kebijakan berbasis identitas yang mengizinkan akses ke atribut DynamoDB. Kebijakan ini memberi izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari AWS API atau AWS CLI. Untuk menggunakan kebijakan ini, ganti *teks cetak miring* dalam contoh kebijakan 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 lebih mempelajari tentang kunci ketentuan DynamoDB, lihat [Menentukan Ketentuan: Menggunakan Kunci Ketentuan](#) dalam 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](#).

```
{
  "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 CLI. Untuk menggunakan kebijakan ini, ganti *teks placeholder yang dicetak miring* dalam kebijakan contoh 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.

```
{
  "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: Melampirkan atau melepaskan volume Amazon EBS ke instans EC2 berdasarkan tag

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan pemilik volume EBS untuk melampirkan atau melepaskan volume EBS 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 CLI. Untuk menggunakan kebijakan ini, ganti *teks placeholder yang dicetak miring* dalam kebijakan contoh dengan informasi Anda sendiri. Lalu, ikuti petunjuk di [buat kebijakan](#) atau [ubah kebijakan](#).

Untuk informasi selengkapnya tentang pembuatan kebijakan IAM untuk mengontrol akses ke sumber daya Amazon EC2, lihat [Mengontrol Akses ke Sumber Daya Amazon EC2](#) dalam Panduan Pengguna Amazon EC2 untuk Instans Linux.

```
{
  "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: Mengizinkan peluncuran instans EC2 dalam subnet tertentu, secara terprogram dan di konsol

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang mengizinkan daftar informasi untuk semua objek EC2 dan meluncurkan instans EC2 dalam subnet tertentu. Kebijakan ini mendefinisikan izin untuk akses terprogram dan konsol. Untuk menggunakan kebijakan ini, ganti *teks placeholder cetak miring* dalam contoh kebijakan dengan informasi Anda sendiri. Lalu, ikuti petunjuk di [buat kebijakan](#) atau [ubah kebijakan](#).

```

{
  "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: Mengizinkan mengelola grup keamanan EC2 dengan pasangan nilai kunci tag tertentu secara terprogram dan di konsol

Contoh ini menunjukkan bagaimana Anda dapat 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 dalam konsol Amazon EC2, tambahkan dan buang aturan ke dalam dan ke luar, dan daftar dan modifikasi deskripsi aturan untuk grup keamanan yang sudah ada dengan `tagDepartment=Test`. Kebijakan ini mendefinisikan izin untuk akses terprogram dan konsol. Untuk menggunakan kebijakan ini, ganti *teks placeholder cetak miring* dalam contoh kebijakan dengan informasi Anda sendiri. Lalu, ikuti petunjuk di [buat kebijakan](#) atau [ubah kebijakan](#).

```
{
  "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:region:111122223333:security-group/*"
    ],
    "Condition": {
      "StringEquals": {
        "aws:ResourceTag/Department": "Test"
      }
    }
  }
}
```

```

    },
    {
      "Effect": "Allow",
      "Action": [
        "ec2:ModifySecurityGroupRules"
      ],
      "Resource": [
        "arn:aws:ec2:region:111122223333:security-group-rule/*"
      ]
    }
  ]
}

```

Amazon EC2: Mengizinkan 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 mengizinkan pengguna IAM memulai atau menghentikan instans EC2, tetapi hanya jika tag `instansOwner` memiliki nilai nama pengguna tersebut. Kebijakan ini mendefinisikan izin untuk akses terprogram dan konsol.

```

{
  "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 instans berdasarkan tanda

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang mengizinkan untuk memulai atau menghentikan instans dengan pasangan tag nilai kunci `Project = DataAnalytics`, tetapi hanya oleh prinsipal dengan pasangan tag nilai kunci `Department = Data`. Kebijakan ini memberi izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari AWS API atau AWS CLI. Untuk menggunakan kebijakan ini, ganti *teks placeholder* dalam contoh kebijakan 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.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "StartStopIfTags",
      "Effect": "Allow",
      "Action": [
        "ec2:StartInstances",
        "ec2:StopInstances"
      ],
      "Resource": "arn:aws:ec2:region:account-id: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 bagaimana Anda dapat membuat kebijakan berbasis identitas yang mengizinkan prinsipal memulai atau menghentikan instans Amazon EC2 ketika tag sumber daya instans dan tag sumber daya instans dan tag prinsipal memiliki nilai yang sama untuk kunci tag sumber daya instans dan tag prinsipal memiliki nilai yang sama untuk kunci tag sumber daya instans dan tag prinsipal memiliki nilai yang sama untuk kunci tag sumber daya instans dan tag prinsipal memiliki nilai yang sama untuk kunci `CostCenter` tag sumber daya instans dan tag prinsipal. Kebijakan ini memberi izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari AWS API atau AWS CLI. Untuk menggunakan kebijakan ini, ganti *teks pengganti cetak* 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.

```

{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "ec2:startInstances",
      "ec2:stopInstances"
    ],
    "Resource": "*",
    "Condition": {"StringEquals": {
      "aws:ResourceTag/CostCenter": "${aws:PrincipalTag/CostCenter"}
    }}
  }
}

```

```
}  
}
```

Amazon EC2: Mengizinkan akses penuh EC2 di dalam Wilayah tertentu, secara terprogram dan di konsol

Contoh ini menunjukkan bagaimana Anda bisa membuat kebijakan berbasis identitas yang mengizinkan akses penuh EC2 di dalam Wilayah tertentu. Kebijakan ini mendefinisikan izin untuk akses terprogram dan konsol. Untuk menggunakan kebijakan ini, ganti *teks pengganti cetak miring* dalam contoh kebijakan 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 ketentuan 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.

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Action": "ec2:*",  
      "Resource": "*",  
      "Effect": "Allow",  
      "Condition": {  
        "StringEquals": {  
          "ec2:Region": "us-east-2"  
        }  
      }  
    }  
  ]  
}
```

Amazon EC2: Mengizinkan memulai atau menghentikan instans EC2 dan memodifikasi grup keamanan, secara terprogram, dan di konsol

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang mengizinkan memulai atau menghentikan instans Amazon EC2 tertentu dan memodifikasi grup keamanan tertentu. Kebijakan ini mendefinisikan izin untuk akses terprogram dan konsol. Untuk

menggunakan kebijakan ini, ganti *teks placeholder cetak miring* dalam contoh kebijakan dengan informasi Anda sendiri. Lalu, ikuti petunjuk di [buat kebijakan](#) atau [ubah kebijakan](#).

```
{
  "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 bagaimana Anda mungkin membuat kebijakan berbasis identitas yang mengizinkan akses penuh ke semua operasi AWS API dalam 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`, lihat [GetSessionToken—kredensial sementara 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 menimpa izin.

Note

Pemeriksaan 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 `BoolIfExists` untuk melihat apakah kuncinya ada sebelum memeriksa nilai. Untuk informasi selengkapnya, lihat [... IfExists operator kondisi](#).

```
{
  "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 kisaran alamat IP

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang membatasi contoh EC2 dengan mengizinkan tindakan tersebut, namun secara tegas menolak akses ketika permintaan datang dari luar kisaran IP yang ditentukan. Kebijakan ini berguna saat alamat IP untuk perusahaan Anda berada dalam cakupan tertentu. Kebijakan ini memberi izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari AWS API atau AWS CLI. Untuk menggunakan kebijakan ini, *ganti teks penghentian* dalam contoh kebijakan dengan informasi informasi Anda sendiri. Lalu, ikuti petunjuk di [buat kebijakan](#) atau [ubah kebijakan](#).

Jika kebijakan ini digunakan bersama-sama dengan kebijakan lain yang memungkinkan `ec2:TerminateInstances` tindakan tersebut (misalnya [AmazonEC2](#) kebijakan yang `FullAccessAWS` dikelola), maka akses ditolak. Hal ini dikarenakan pernyataan penolakan yang eksplisit lebih diutamakan daripada pernyataan yang membolehkan. Untuk informasi selengkapnya, lihat [the section called "Menentukan apakah permintaan diizinkan atau ditolak dalam sebuah akun."](#)

Important

Kunci kondisi `aws:SourceIp` menolak akses ke layanan AWS, seperti AWS CloudFormation, yang melakukan panggilan atas nama Anda. Untuk informasi selengkapnya tentang penggunaan kunci kondisi `aws:SourceIp`, lihat [AWS kunci konteks kondisi global](#).

```

{
  "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: Mengakses API simulator kebijakan: Mengakses API simulator kebijakan Mengakses API simulator kebijakan

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis kebijakan yang mengizinkan API simulator kebijakan untuk kebijakan yang melekat ke pengguna, grup, atau peran dalam akun AWS. Kebijakan ini juga mengizinkan akses untuk menyimulasikan kebijakan yang kurang sensitif untuk diteruskan ke API sebagai strings. Kebijakan ini memberi izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari AWS API yang atau AWS CLI.

```
{
  "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 untuk menyimulasikan kebijakan yang melekat ke pengguna, grup, atau peran dalam saat ini Akun AWS, lihat [IAM::: Mengakses konsol simulator kebijakan](#).

IAM::: Mengakses konsol simulator kebijakan

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan yang mengizinkan penggunaan konsol simulator kebijakan untuk kebijakan yang melekat ke pengguna, grup, atau peran dalam saat ini Akun AWS. Kebijakan ini memberi izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari AWS API atau AWS CLI.

Anda dapat mengakses konsol IAM Policy Simulator di: <https://policysim.aws.amazon.com/>

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "iam:GetGroup",
        "iam:GetGroupPolicy",
        "iam:GetPolicy",
        "iam:GetPolicyVersion",
        "iam:GetRole",
        "iam:GetRolePolicy",
        "iam:GetUser",
        "iam:GetUserPolicy",
        "iam:ListAttachedGroupPolicies",
        "iam:ListAttachedRolePolicies",
        "iam:ListAttachedUserPolicies",
        "iam:ListGroups",
        "iam:ListGroupPolicies",
        "iam:ListGroupsForUser",
        "iam:ListRolePolicies",
        "iam:ListRoles",
        "iam:ListUserPolicies",
        "iam:ListUsers"
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
}
```

```

    }
  ]
}

```

IAM: Ambil peran yang memiliki tanda tertentu

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang mengizinkan pengguna IAM mengambil peran dengan tag pasangan nilai-kunci `Project = ExampleCorpABC`. Kebijakan ini memberi izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari AWS API atau AWS CLI. Untuk menggunakan kebijakan ini, ganti *teks cetak miring* dalam contoh kebijakan 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 [Menyediakan akses ke pengguna IAM di pengguna lain Akun AWS yang Anda miliki](#).

```

{
  "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 mengizinkan akses penuh ke beberapa layanan dan membatasi akses swakelola di IAM. Ia juga

menolak akses ke bucket logs Amazon S3 atau contoh `i-1234567890abcdef0` Amazon EC2. Kebijakan ini mendefinisikan izin untuk akses terprogram dan konsol. Untuk menggunakan kebijakan ini, ganti *teks placeholder cetak miring* di contoh kebijakan 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 AWS Management Console. 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. Saat kebijakan digunakan sebagai batasan izin pada pengguna, meskipun ada kebijakan lain yang dilampirkan ke pengguna mengizinkan tindakan tersebut, AWS menolak permintaan tersebut.

```
{
  "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 bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan menambahkan kunci tagDepartment dengan nilai tagMarketing, Development, atau Quality Assurance untuk pengguna IAM. Pengguna tersebut harus sudah menyertakan tag key—value pair JobFunction = manager. Anda dapat menggunakan kebijakan ini untuk mewajibkan satu manajer hanya dimiliki oleh satu dari tiga departemen. Kebijakan ini mendefinisikan izin untuk akses terprogram dan konsol. Untuk menggunakan kebijakan ini, ganti *teks placeholder miring* dalam kebijakan contoh 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 [Kunci konteks multivaluasi](#).

```
{
  "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 menambahkan hanya kunci tagCostCenter dan baik nilai tagA-123 atau nilai tagB-456 untuk setiap pengguna IAM atau peran. Anda dapat menggunakan kebijakan ini untuk membatasi penandaan ke kunci tanda tertentu dan serangkaian nilai tanda. Kebijakan ini mendefinisikan izin untuk akses terprogram dan konsol. Untuk menggunakan kebijakan ini, ganti *teks placeholder miring* dalam kebijakan contoh 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 [Kunci konteks multivaluasi](#).

```
{
  "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 `Department` dan `JobFunction` kunci tag. Kunci tanda `Department` harus memiliki nilai tanda `Development` atau `QualityAssurance`. Kunci tanda `JobFunction` harus memiliki nilai 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 `AWSAPI` atau `AWS CLI`. Untuk menggunakan kebijakan ini, ganti *teks placeholder miring* dalam kebijakan contoh 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 [Kunci konteks multivaluasi](#).

```
{
  "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: Buat dan ambil IAM

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan kredensial yang mengizinkan pengguna untuk membuat dan mengunduh laporan yang mencatat semua pengguna IAM di merekaAkun AWS. Laporan ini termasuk status dari kredensial pengguna, termasuk kata sandi, access key, perangkat MFA, dan sertifikat tanda tangan. Kebijakan ini memberi izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dariAWS API atauAWS CLI.

Untuk informasi selengkapnya tentang laporan kredensial, lihat [Mendapatkan laporan kredensi untuk Anda Akun AWS](#).

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

IAM: Memungkinkan pengelolaan keanggotaan grup secara terprogram dan di konsol

Contoh ini menunjukkan bagaimana Anda bisa membuat kebijakan berbasis identitas yang mengizinkan pembaruan keanggotaan grup yang dipanggil `MarketingTeam`. Kebijakan ini mendefinisikan izin untuk akses terprogram dan konsol. Untuk menggunakan kebijakan ini, ganti *teks cetak miring* dalam contoh kebijakan 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 AWS Management Console. 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.

```
{
  "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: Mengelola tag tertentu

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan penambahan dan penghapusan tag IAM dengan kunci tagDepartment dari entitas IAM (pengguna dan peran). Kebijakan ini tidak membatasi nilaiDepartment tag. Kebijakan ini memberi izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari AWS API atau AWS CLI. Untuk menggunakan kebijakan ini, ganti *teks placeholder cetak miring* dalam contoh kebijakan dengan informasi Anda sendiri. Lalu, ikuti petunjuk di [buat kebijakan](#) atau [ubah kebijakan](#).

```

{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "iam:TagUser",
      "iam:TagRole",
      "iam:UntagUser",
      "iam:UntagRole"
    ],
    "Resource": "*",
    "Condition": {"ForAllValues:StringEquals": {"aws:TagKeys": "Department"}}
  }
}

```

IAM: Teruskan peran IAM ke AWS layanan khusus

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang mengizinkan penerusan peran layanan IAM apa pun ke CloudWatch layanan Amazon Amazon. Kebijakan ini memberi izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari AWS API atau AWS CLI. Untuk menggunakan kebijakan ini, ganti *teks placeholder cetak*

piring dalam contoh kebijakan 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 mengizinkan Amazon Amazon CloudWatch mengambil peran yang Anda teruskan, Anda harus menentukan `cloudwatch.amazonaws.com` layanan utama sebagai prinsipal dalam kebijakan kepercayaan dari peran Anda. Prinsipal 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 [Memberi izin kepada pengguna untuk meneruskan peran ke layanan AWS](#).

```
{
  "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 pelaporan

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan pengguna `Get` atau pengguna `List`. Saat pengguna bekerja dengan konsol, konsol membuat permintaan ke IAM untuk mencantumkan grup, pengguna, peran, dan kebijakan, serta membuat 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 dengan `Get`, seperti `getUser` dan `getRole`.

Wildcard berguna jika tipe entitas baru ditambahkan ke IAM di masa future. 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-](#)

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

IAM: Mengizinkan akses hanya-

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan pengguna IAM untuk melakukan tindakan IAM yang dimulai dengan `stringGet`, `List`, atau `Generate`. Saat pengguna bekerja dengan konsol IAM, konsol membuat permintaan untuk mencantumkan grup, pengguna, peran, dan kebijakan, dan untuk membuat 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 dengan `Get`, seperti `getUser` dan `getRole`. Menggunakan wildcard bermanfaat, terutama jika tipe entitas baru ditambahkan ke IAM di masa future. 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 pelaporan](#).

```
{
  "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 dalam konsol

Contoh ini menunjukkan bagaimana Anda bisa membuat kebijakan berbasis identitas yang memungkinkan pengguna IAM tertentu untuk mengelola `AllUsers` grup. Kebijakan ini mendefinisikan izin untuk akses terprogram dan konsol. Untuk menggunakan kebijakan ini, ganti *teks placeholder cetak miring* dalam contoh kebijakan 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 tipe sumber daya yang dapat Anda tentukan untuk tindakan IAM, lihat [Tindakan, Sumber Daya, dan Kunci Kondisi untuk AWS Identity and Access Management](#).
- Pernyataan `LimitGroupManagementAccessToSpecificUsers` itu menolak pengguna dengan nama tertentu mengakses untuk menulis dan pengelolaan izin tindakan grup. 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 tab Editor visual. Untuk informasi

selengkapnya tentang tingkat akses. lihat [Tindakan, Sumber Daya, dan Kunci Kondisi untuk AWS Identity and Access Management](#).

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowAllUsersToListAllGroups",
      "Effect": "Allow",
      "Action": "iam:ListGroup",
      "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: Mengizinkan pengaturan persyaratan kata sandi secara terprogram dan di konsol

Contoh ini menunjukkan bagaimana Anda bisa membuat kebijakan berbasis identitas yang memungkinkan pengguna 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 mendefinisikan izin untuk akses terprogram dan konsol.

Untuk mempelajari bagaimana mengatur persyaratan kata sandi akun untuk akun Anda, lihat [Mengatur kebijakan kata sandi akun untuk pengguna IAM](#).

```

{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "iam:GetAccountPasswordPolicy",
      "iam:UpdateAccountPasswordPolicy"
    ],
    "Resource": "*"
  }
}

```

IAM: Mengakses API simulator kebijakan berdasarkan jalur pengguna

Contoh ini menunjukkan cara membuat kebijakan berbasis kebijakan yang memungkinkan penggunaan API simulator kebijakan hanya untuk pengguna yang memiliki jalur tersebut `Department/Development`. Kebijakan ini memberi izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari AWS API atau AWS CLI. Untuk menggunakan kebijakan ini, ganti *teks placeholder yang dicetak miring* dalam kebijakan contoh dengan informasi Anda sendiri. Lalu, ikuti petunjuk di [buat kebijakan](#) atau [ubah kebijakan](#).

```

{
  "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: Mengakses konsol simulator kebijakan berdasarkan jalur](#).

IAM: Mengakses konsol simulator kebijakan berdasarkan jalur

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan penggunaan konsol simulator kebijakan hanya untuk pengguna yang memiliki jalur `Department/Development`. Kebijakan ini memberi izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari AWS API atau AWS CLI. Untuk menggunakan kebijakan ini, ganti *teks placeholder cetak miring* dalam kebijakan contoh dengan informasi Anda sendiri. Lalu, ikuti petunjuk di [buat kebijakan](#) atau [edit kebijakan](#).

Anda dapat mengakses konsol Simulator Kebijakan IAM di: <https://policysim.aws.amazon.com/>

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "iam:GetPolicy",
        "iam:GetUserPolicy"
      ],
      "Effect": "Allow",
      "Resource": "*"
    },
    {
      "Action": [
        "iam:GetUser",
```

```

        "iam:ListAttachedUserPolicies",
        "iam:ListGroupsWithUser",
        "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 bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan pengguna IAM untuk mengelola sendiri [otentikasi multi-faktor \(MFA\)](#) perangkat. Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari AWS API atau AWS CLI.

Note

Jika pengguna IAM dengan kebijakan ini tidak diautentikasi MFA, kebijakan ini menolak akses ke semua AWS tindakan kecuali yang diperlukan untuk mengotentikasi menggunakan MFA. Jika Anda menambahkan izin ini untuk pengguna yang masuk ke AWS, mereka mungkin perlu keluar dan masuk kembali untuk melihat perubahan ini.

```

{
  "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 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.

```
{
  "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 yang terakhir diakses untuk kebijakan Organizations yang terakhir diakses untuk

Contoh ini menunjukkan bagaimana Anda bisa membuat kebijakan berbasis identitas yang memungkinkan untuk kebijakan Organizations tertentu. Kebijakan ini memungkinkan pengambilan data untuk kebijakan kendali layanan (SCP) dengan `p-policy123` ID. Orang yang membuat dan melihat laporan harus diautentikasi menggunakan kredensial akun AWS Organizations manajemen. Kebijakan ini memungkinkan pemohon mengambil data bagi entitas Organisasi di dalam organisasinya. Kebijakan ini mendefinisikan izin untuk akses terprogram dan konsol. Untuk menggunakan kebijakan ini, ganti *teks placeholder yang dicetak miring* dalam contoh. 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 [Menyaring izin di AWS menggunakan informasi yang terakhir diakses](#).

```
{
  "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 ke pengguna, grup atau peran IAM.

Contoh ini menunjukkan bagaimana Anda bisa membuat kebijakan berbasis identitas yang membatasi kebijakan yang dikelola pelanggan dan AWS dikelola yang dapat diterapkan ke pengguna, grup, atau peran IAM. Kebijakan ini memberi izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari AWS API atau AWS CLI. Untuk menggunakan kebijakan ini, ganti *teks placeholder cetak miring* dalam contoh kebijakan dengan informasi Anda sendiri. Lalu, ikuti petunjuk di [buat kebijakan](#) atau [ubah kebijakan](#).

```
{
  "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 AWS kebijakan IAM terkelola

Menggunakan `aws:ResourceAccount` dalam 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 memungkinkan AWS kebijakan IAM terkelola. Akun yang dikelola layanan di luar AWS Organizations memiliki Kebijakan IAM Terkelola. Ada empat tindakan IAM yang mencantumkan dan

mengambil AWS kebijakan -managed. Gunakan tindakan ini di [NotAction](#) elemen pernyataan. `AllowAccessToS3ResourcesInSpecificAccountsAndSpecificService1` dalam kebijakan.

```
{
  "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: Memungkinkan fungsi Lambda mengakses tabel Amazon DynamoDB

Contoh ini menunjukkan bagaimana Anda bisa membuat kebijakan berbasis identitas yang mengizinkan akses baca dan tulis ke tabel Amazon DynamoDB tertentu. Kebijakan ini juga mengizinkan penulisan fail CloudWatch log ke Logs. Untuk menggunakan kebijakan ini, ganti *teks placeholder cetak miring* dalam contoh kebijakan 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 tersebut harus mencakup AWS Lambda sebagai prinsipal dalam kebijakan kepercayaan. Untuk rincian tentang bagaimana menggunakan kebijakan ini, lihat [Cara](#)

Membuat Kebijakan AWS IAM untuk Memberi AWS Lambda Tabel DynamoDB Amazon DynamoDB di Blog AWS Security.

```
{
  "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": "*"
    }
  ]
}
```



```

        "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 tanda mengakses sumber daya RDS yang telah mereka tandai sepenuhnya

Contoh ini menunjukkan bagaimana Anda bisa membuat kebijakan berbasis identitas yang memungkinkan pemilik tag mengakses sumber daya RDS yang mereka tandai sepenuhnya. Kebijakan ini memberi izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari AWS API atau AWS CLI.

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "rds:Describe*",
        "rds:List*"
      ],
      "Effect": "Allow",
      "Resource": "*"
    },
    {
      "Action": [
        "rds>DeleteDBInstance",
        "rds:RebootDBInstance",
        "rds:ModifyDBInstance"
      ],
      "Effect": "Deny",
      "Resource": "*"
    }
  ]
}

```



```

    ],
    "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 dalam bucket mereka

Contoh ini menunjukkan bagaimana Anda bisa membuat kebijakan berbasis identitas yang memungkinkan pengguna Amazon Cognito untuk mengakses obyek di bucket S3 tertentu. Kebijakan ini memungkinkan akses hanya ke obyek dengan nama yang mencantumkan `cognito`, nama aplikasi, dan ID pengguna federasi, yang diwakili oleh variabel `${cognito-`

identity.amazonaws.com:sub} Kebijakan ini memberi izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari AWS API atau AWS CLI. Untuk menggunakan kebijakan ini, ganti *teks placeholder cetak miring* dalam contoh kebijakan 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.

```
{
  "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. Kolam identitas memungkinkan Anda memberi pengguna Anda akses ke layanan AWS lainnya. 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: Mengizinkan pengguna gabungan mengakses direktori rumah S3 mereka, secara terprogram dan di konsol

Contoh ini menunjukkan bagaimana Anda bisa membuat kebijakan berbasis identitas yang memungkinkan pengguna federasi untuk mengakses bucket direktori rumahnya sendiri di S3. Direktori rumah adalah bucket yang mencakup home folder dan folders untuk pengguna federasi individual. Kebijakan ini mendefinisikan izin untuk akses terprogram dan konsol. Untuk menggunakan kebijakan ini, ganti *teks placeholder cetak miring* dalam contoh kebijakan 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`. Bagian `role-id` ID pengguna gabungan adalah pengidentifikasi unik yang diberikan ke peran pengguna gabungan selama pembuatan. Untuk informasi selengkapnya, lihat [Pengidentifikasi unik](#). Ini `specified-name` adalah [RoleSessionName parameter](#) yang diteruskan ke `AssumeRoleWithWebIdentity` permintaan ketika pengguna federasi mengambil perannya.

Anda dapat melihat ID peran menggunakan perintah AWS CLI `aws iam get-role --role-name specified-name`. Misalnya, bayangkan Anda menentukan nama John yang mudah diingat dan CLI mengembalikan ID peran `AR0AXXT2NJT7D3SIQN7Z6`. Dalam kasus ini, ID pengguna gabungan adalah `AR0AXXT2NJT7D3SIQN7Z6:John`. Kebijakan ini selanjutnya memungkinkan pengguna gabungan John mengakses bucket Amazon S3 dengan prefiks `AR0AXXT2NJT7D3SIQN7Z6:John`.

```
{
  "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:::bucket-name",
  "Condition": {
    "StringLike": {
      "s3:prefix": [
        "",
        "home/",
        "home/${aws:userid}/*"
      ]
    }
  }
},
{
  "Effect": "Allow",
  "Action": "s3:*",
  "Resource": [
    "arn:aws:s3:::bucket-name/home/${aws:userid}",
    "arn:aws:s3:::bucket-name/home/${aws:userid}/*"
  ]
}
]
```

Amazon S3: Mengakses bucket S3, tetapi bucket produksi menolak tanpa MFA terkini

Contoh ini menunjukkan bagaimana Anda bisa membuat kebijakan berbasis identitas yang memungkinkan administrator Amazon S3 mengakses bucket mana pun, termasuk memperbarui, menambahkan, dan menghapus obyek. Namun, ini secara eksplisit menolak akses ke 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 secara terprogram menggunakan AWS CLI atau AWS API. Untuk menggunakan kebijakan ini, ganti *teks placeholder cetak miring* dalam contoh kebijakan dengan informasi Anda sendiri. Lalu, ikuti petunjuk di [buat kebijakan](#) atau [ubah kebijakan](#).

Kebijakan ini tidak pernah mengizinkan akses terprogram ke bucket `Production` 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 `Production` bucket secara terprogram, administrator S3 harus menggunakan kredensial sementara yang dibuat dalam 30 menit terakhir menggunakan [GetSessionToken](#) operasi API.

```
{
  "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:::Production/*",
        "arn:aws:s3:::Production"
    ],
    "Condition": {
        "NumericGreaterThanIfExists": {"aws:MultiFactorAuthAge": "1800"}
    }
}
]
}

```

Amazon S3: Mengizinkan akses pengguna IAM untuk mengakses direktori rumah S3 mereka, secara terprogram dan di konsol

Contoh ini menunjukkan bagaimana Anda bisa membuat kebijakan berbasis identitas yang memungkinkan pengguna IAM untuk mengakses bucket direktori rumahnya sendiri di S3. Direktori rumah adalah bucket yang mencakup home folder dan folders untuk pengguna individual. Kebijakan ini mendefinisikan izin untuk akses terprogram dan konsol. Untuk menggunakan kebijakan ini, ganti *teks tempat cetak miring* dalam contoh kebijakan dengan informasi Anda sendiri. Lalu, ikuti petunjuk di [buat kebijakan](#) atau [ubah kebijakan](#).

Kebijakan IAM ini tidak akan berfungsi saat menggunakan peran IAM karena `aws:username` variabel tidak tersedia ketika menggunakan peran IAM ini tidak akan berfungsi saat menggunakan peran IAM ini tidak akan berfungsi saat menggunakan peran IAM ini Untuk detail tentang nilai kunci prinsipal, lihat [Nilai-nilai kunci utama](#).

```

{
    "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:::bucket-name",
  "Condition": {
    "StringLike": {
      "s3:prefix": [
        "",
        "home/",
        "home/${aws:username}/*"
      ]
    }
  }
},
{
  "Effect": "Allow",
  "Action": "s3:*",
  "Resource": [
    "arn:aws:s3:::bucket-name/home/${aws:username}",
    "arn:aws:s3:::bucket-name/home/${aws:username}/*"
  ]
}
]
```


Amazon S3 tertentu

Contoh ini menunjukkan bagaimana Anda bisa membuat kebijakan berbasis identitas yang membatasi mengelola bucket tertentu. Kebijakan ini memberikan izin untuk melakukan semua tindakan Amazon S3, tetapi menolak akses ke setiap tindakan Layanan AWS kecuali Amazon S3. Lihat contoh berikut. Menurut kebijakan ini, Anda hanya dapat mengakses tindakan Amazon S3 yang dapat Anda lakukan pada bucket S3 atau sumber daya objek S3. Kebijakan ini memberi izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari AWS API atau AWS CLI. Untuk menggunakan kebijakan ini, ganti *teks placeholder cetak miring* dalam contoh kebijakan dengan informasi Anda sendiri. Lalu, ikuti petunjuk di [buat kebijakan](#) atau [ubah kebijakan](#).

Jika kebijakan ini digunakan bersama-sama dengan kebijakan lain (misalnya [AmazonS3FullAccess](#) atau [AmazonEC2 kebijakan yang FullAccess AWS dikelola 2](#)) yang memungkinkan tindakan ditolak oleh kebijakan ini, akses akan ditolak. Hal ini dikarenakan pernyataan penolakan yang eksplisit lebih diutamakan daripada pernyataan yang membolehkan. Untuk informasi selengkapnya, lihat [the section called "Menentukan apakah permintaan diizinkan atau ditolak dalam sebuah akun."](#)

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.

```
{
  "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/*"
    ]
}
}
```

Amazon S3: Mengizinkan akses baca dan tulis bagi obyek dalam bucket S3

Contoh ini menunjukkan bagaimana Anda bisa membuat kebijakan berbasis identitas yang memungkinkan `Read` dan `Write` mengakses obyek di bucket S3 tertentu. Kebijakan ini memberi izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari AWS API atau AWS CLI. Untuk menggunakan kebijakan ini, ganti *teks placeholder dicetak miring* dalam contoh kebijakan 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".

```
{
  "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 dalam Bucket S3, secara terprogram dan di konsol](#).

Amazon S3: Memungkinkan akses baca dan tulis ke objek dalam Bucket S3, secara terprogram dan di konsol

Contoh ini menunjukkan bagaimana Anda bisa membuat kebijakan berbasis identitas yang memungkinkan Read dan Write mengakses obyek di bucket S3 tertentu. Kebijakan ini mendefinisikan izin untuk akses terprogram dan konsol. Untuk menggunakan kebijakan ini, ganti *teks placeholder cetak miring* dalam contoh kebijakan 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".

```
{
  "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:::bucket-name"]
  },
  {
    "Sid": "AllObjectActions",
    "Effect": "Allow",
    "Action": "s3:*Object",
    "Resource": ["arn:aws:s3:::bucket-name/*"]
  }
]
```

Mengelola 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.

Lihat sumber daya ini untuk rinciannya:

- Untuk informasi lebih lanjut tentang tipe kebijakan IAM yang berbeda, lihat [Kebijakan dan Izin di IAM](#).
- Untuk informasi umum tentang penggunaan kebijakan di dalam IAM, lihat [Manajemen akses untuk sumber daya AWS](#).
- 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. Untuk informasi selengkapnya, lihat [IAM dan AWS STS kuota](#).

Topik

- [Membuat kebijakan IAM](#)
- [Memvalidasi kebijakan IAM](#)
- [Hasilkan kebijakan berdasarkan aktivitas akses](#)
- [Menguji kebijakan IAM dengan simulator kebijakan IAM](#)
- [Menambahkan dan menghapus izin identitas IAM](#)

- [Peningkatan versi IAM](#)
- [Menyunting Kebijakan IAM](#)
- [Menghapus kebijakan IAM](#)
- [Menyaring izin di AWS menggunakan informasi yang terakhir diakses](#)

Membuat kebijakan IAM

Sebuah [kebijakan](#) adalah entitas yang, saat dilampirkan dengan sebuah identitas atau sumber daya, menjelaskan izinnya. Anda dapat menggunakan AWS Management Console, AWS CLI, atau AWS API untuk membuat kebijakan yang dikelola pelanggan di IAM. Kebijakan yang selaras Akun AWS. Anda lalu dapat melampirkan kebijakan itu ke identitas (pengguna, grup, dan dan peran) di Anda Akun AWS.

Kebijakan yang terlampir pada identitas di IAM dikenal sebagai kebijakan berbasis-identitas. Kebijakan berbasis identitas dapat mencakup kebijakan terkelola AWS, kebijakan terkelola pelanggan, dan kebijakan inline. Kebijakan terkelola AWS dibuat dan dikelola oleh AWS. Anda dapat menggunakannya, tapi tidak bisa mengelolanya. Kebijakan yang selaras adalah kebijakan yang Anda buat dan tanamkan langsung ke grup, pengguna, atau peran IAM. Kebijakan yang selaras tidak bisa dipakai kembali untuk identitas lain atau dikelola di luar identitas dimana ia berada. Untuk informasi selengkapnya, lihat [Menambahkan dan menghapus izin identitas IAM](#).

Menggunakan kebijakan yang selaras. Juga paling baik menggunakan kebijakan yang dikelola pelanggan dan bukan kebijakan terkelola AWS. AWS Kebijakan terkelola biasanya menyediakan izin administratif atau hanya-baca yang luas. Untuk keamanan tertinggi, [berikan privilese terkecil](#), hanya mengizinkan seperlunya untuk melakukan tugas tertentu.

Saat membuat atau mengedit kebijakan IAM, AWS dapat secara otomatis melakukan validasi kebijakan untuk membantu Anda membuat kebijakan yang efektif dengan mempertimbangkan hak istimewa. Di AWS Management Console, IAM mengidentifikasi kesalahan sintaks JSON, sementara IAM Access Analyzer memberikan pemeriksaan kebijakan tambahan dengan rekomendasi untuk membantu Anda lebih menyempurnakan kebijakan Anda. Untuk mempelajari selengkapnya tentang validasi kebijakan, lihat [Memvalidasi kebijakan IAM](#). Untuk mempelajari lebih lanjut tentang pemeriksaan kebijakan IAM Access Analyzer dan rekomendasi yang dapat ditindaklanjuti, lihat [validasi kebijakan IAM Access Analyzer](#).

Anda dapat menggunakan AWS Management Console, AWS CLI, atau AWS API untuk membuat kebijakan yang dikelola pelanggan di IAM. Untuk informasi selengkapnya tentang menggunakan AWS

CloudFormation template untuk menambah atau memperbarui kebijakan, lihat [referensi jenisAWS Identity and Access Management sumber daya](#) di PanduanAWS CloudFormation Pengguna.

Topik

- [Membuat kebijakan IAM \(konsol\)](#)
- [Membuat kebijakan IAM \(AWS CLI\)](#)
- [Membuat kebijakan IAM \(AWSAPI\)](#)

Membuat kebijakan IAM (konsol)

Sebuah [kebijakan](#) adalah entitas yang, saat dilampirkan dengan sebuah identitas atau sumber daya, menjelaskan izinnya. Anda dapat menggunakanAWS Management Consoleuntuk membuatkebijakan yang dikelola pelanggan di IAM. Kebijakan yang dikelola pelanggan adalah kebijakan mandiri yang Anda kelola sendiriAkun AWS. Anda kemudian dapat melampirkan kebijakan ke identitas (pengguna, grup, dan peran) diAkun AWS.

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 yang dikelola pelanggan di AWS Management Console 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 bisa mengimpor kebijakan terkelola AWS atau kebijakan yang dikelola pelanggan yang sebelumnya telah Anda buat.

Jumlah dan ukuran sumber daya IAM dalamAWSakun 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 JSON pilihan. 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 JSON pilihan 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 memberikan 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 IAM](#). Untuk informasi tentang bahasa kebijakan IAM, lihat [Referensi kebijakan IAM JSON](#).

Untuk menggunakan editor kebijakan JSON untuk membuat kebijakan

1. Masuk ke AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Di panel navigasi sebelah kiri, pilih Kebijakan.
3. Pilih Buat kebijakan.
4. Dalam Editor kebijakan bagian, pilih JSON pilihan.
5. Ketik atau tempel dokumen kebijakan JSON. Untuk rincian bahasa kebijakan IAM, lihat [Referensi kebijakan IAM JSON](#).
6. Mengatasi peringatan keamanan, kesalahan, atau peringatan umum yang dihasilkan selama [validasi kebijakan](#), dan kemudian pilih Berikutnya.

Note

Anda dapat beralih di antara Visual dan JSON opsi editor kapan saja. Namun, jika Anda membuat perubahan atau memilih Berikutnya di dalam Visual editor, IAM mungkin

merestrukturisasi kebijakan Anda untuk mengoptimalkannya untuk editor visual. Untuk informasi selengkapnya, lihat [Restrukturisasi kebijakan](#).

7. (Opsional) Saat Anda membuat atau mengedit kebijakan di AWS Management Console, Anda dapat membuat template kebijakan JSON atau YAKL yang dapat Anda gunakan AWS CloudFormation template.

Untuk melakukan ini, di Editor kebijakan pilih Tindakan, dan kemudian pilih Menghasilkan CloudFormation template. Untuk mempelajari lebih lanjut tentang AWS CloudFormation, lihat [AWS Identity and Access Management referensi tipe sumber daya](#) di dalam AWS CloudFormation Panduan Pengguna.

8. Setelah selesai menambahkan izin ke kebijakan, pilih Berikutnya.
9. Pada Tinjau dan buatkan halaman, ketikkan Nama Kebijakan dan Deskripsi (opsional) untuk kebijakan yang Anda buat. Ulasan 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 [Menandai sumber daya IAM](#).
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 AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Di panel navigasi sebelah kiri, pilih Kebijakan.
3. Pilih Buat kebijakan.
4. Dalam Editor kebijakan bagian, temukan Pilih layanan bagian, dan kemudian 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 memilihTambahkan lebih banyak izin.

5. DalamTindakan 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 mendukung[izin tingkat sumber daya](#), maka editor visual akan mendaftarkan sumber daya tersebut. Kemudian Anda bisa memperluas Sumber Daya untuk menentukan sumber daya bagi kebijakan Anda.

Anda dapat menentukan sumber daya dengan cara berikut:

- PilihTambahkan ARNuntuk menentukan sumber daya berdasarkan Amazon Resource Names (ARN) mereka. Anda dapat menggunakan editor ARN visual atau mendaftarkan ARN secara manual. Untuk informasi lebih lanjut tentang sintaks ARN, lihat [Amazon Resource Name \(ARN\)](#) pada Panduan Referensi Umum AWS. Untuk informasi tentang penggunaan ARN dalam elemen Resource kebijakan, lihat [Elemen kebijakan IAM JSON: Resource](#).

- Pilih Apa saja di akun inidi 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 Kondisi permintaan -pilihan untuk menambahkan kondisi pada 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 IAM](#) di dalam Referensi Otorisasi Layanan.


Anda bisa memilih kondisi dengan cara berikut:

- Gunakan kotak centang untuk memilih kondisi yang biasa dipakai.
- 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 Menambahkan. 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: Condition](#) di [Referensi kebijakan IAM JSON](#).

8. Untuk menambahkan lebih banyak blok izin, pilih Tambahkan lebih banyak izin. Untuk setiap blok, ulangi langkah 2 sampai 5.

 Note

Anda dapat beralih di antara Visual dan JSON opsi editor kapan saja. Namun, jika Anda membuat perubahan atau memilih Berikutnya di dalam Visual editor, IAM mungkin

merestrukturisasi kebijakan Anda untuk mengoptimalkannya untuk editor visual. Untuk informasi selengkapnya, lihat [Restrukturisasi kebijakan](#).

9. (Opsional) Saat Anda membuat atau mengedit kebijakan di AWS Management Console, Anda dapat membuat template kebijakan JSON atau YAKL yang dapat Anda gunakan AWS CloudFormation template.

Untuk melakukan ini, di Editor kebijakan pilih Tindakan, dan kemudian pilih Menghasilkan CloudFormation template. Untuk mempelajari lebih lanjut tentang AWS CloudFormation, lihat [AWS Identity and Access Management referensi tipe sumber daya](#) di dalam AWS CloudFormation Panduan Pengguna.

10. Setelah selesai menambahkan izin ke kebijakan, pilih Berikutnya.
11. Pada Tinjau dan buat halaman, ketikkan 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 dimaksudkan.
12. (Opsional) Tambahkan metadata ke kebijakan dengan melampirkan tanda sebagai pasangan nilai kunci. Untuk informasi selengkapnya tentang menggunakan tanda di IAM, lihat [Menandai sumber daya IAM](#).
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 AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Di panel navigasi sebelah kiri, pilih Kebijakan.

3. Pilih Buat kebijakan.
4. Dalam Editor kebijakan, pilih Visual dan kemudian di sisi kanan halaman, pilih Tindakan maka pilihlah Kebijakan impor.
5. Dalam Kebijakan impor window, 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. Kemudian pilih Selanjutnya.

 Note

Anda dapat beralih di antara Visual dan JSON opsi editor kapan saja. Namun, jika Anda membuat perubahan atau memilih Berikutnya di dalam Visual editor, IAM mungkin merestrukturisasi kebijakan Anda untuk mengoptimalkannya untuk editor visual. Untuk informasi selengkapnya, lihat [Restrukturisasi kebijakan](#).

8. Pada Tinjau dan buat halaman, ketik Nama Kebijakan dan Deskripsi (opsional) untuk kebijakan yang Anda buat. Anda tidak bisa mengedit pengaturan kemudian. Tinjaulzin yang ditentukan dalam kebijakan inidan kemudian pilih Buat kebijakan untuk menyimpan pekerjaan Anda.

Untuk mengimpor kebijakan terkelola yang ada di JSON editor

1. Masuk ke AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Di panel navigasi sebelah kiri, pilih Kebijakan.
3. Pilih Buat kebijakan.
4. Dalam Editor kebijakan bagian, pilih JSON pilihan, dan kemudian di sisi kanan halaman, pilih Tindakan maka pilihlah Kebijakan impor.
5. Dalam Kebijakan impor window, 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. Mengatasi peringatan keamanan, kesalahan, atau peringatan umum yang dihasilkan selama [validasi kebijakan](#), dan kemudian pilih Berikutnya. Sesuaikan kebijakan Anda di JSON, atau pilih Visual editor (Editor visual). Kemudian pilih Selanjutnya.

Note

Anda dapat beralih di antara Visual dan JSON opsi editor kapan saja. Namun, jika Anda membuat perubahan atau memilih Berikutnya di dalam Visual editor, IAM mungkin merestrukturisasi kebijakan Anda untuk mengoptimalkannya untuk editor visual. Untuk informasi selengkapnya, lihat [Restrukturisasi kebijakan](#).

8. Pada Tinjau dan buat halaman, ketikkan 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 dan kemudian pilih Buat kebijakan untuk menyimpan pekerjaan 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 IAM (AWS CLI)

Sebuah [kebijakan](#) 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 yang Anda kelola di Akun AWS. Sebagai [praktik terbaik](#), kami menyarankan Anda menggunakan IAM Access Analyzer untuk memvalidasi kebijakan IAM Anda untuk memastikan izin yang aman dan fungsional. Dengan [memvalidasi kebijakan Anda](#), Anda dapat mengatasi kesalahan atau rekomendasi sebelum Anda melampirkan kebijakan itu ke identitas (pengguna, grup, dan peran) di Akun AWS.

Jumlah dan ukuran sumber daya IAM di Akun AWS terbatas. Untuk informasi selengkapnya, lihat [IAM dan AWS STS kuota](#).

Membuat kebijakan IAM (AWS CLI)

Anda dapat membuat kebijakan IAM yang dikelola pelanggan atau kebijakan yang selaras dengan 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:

- [memvalidasi-kebijakan](#)

Membuat kebijakan IAM (AWSAPI)

Sebuah [kebijakan](#) adalah entitas yang, saat dilampirkan dengan sebuah identitas atau sumber daya, menjelaskan izinnya. Anda dapat menggunakan AWS API itu ke kebijakan yang dikelola pelanggan di IAM. Kebijakan yang dikelola pelanggan adalah kebijakan yang berdiri sendiri yang Anda kelola di Akun AWS. Sebagai [praktik terbaik](#), kami menyarankan Anda menggunakan IAM Access Analyzer untuk memvalidasi kebijakan IAM Anda untuk memastikan izin yang aman dan fungsional. Dengan [memvalidasi kebijakan Anda](#) dapat mengatasi kesalahan atau rekomendasi sebelum melampirkan kebijakan itu ke identitas (pengguna, grup, dan peran) di akun Anda Akun AWS.

Jumlah dan ukuran sumber daya IAM di AWS akun terbatas. Untuk informasi selengkapnya, lihat [IAM dan AWS STS kuota](#).

Membuat kebijakan IAM (AWSAPI)

Anda dapat membuat kebijakan IAM yang dikelola pelanggan atau kebijakan inline menggunakan AWS API.

Untuk membuat kebijakan yang dikelola pelanggan (API AWS)

Panggil operasi berikut ini:

- [CreatePolicy](#)

Untuk membuat kebijakan inline untuk identitas IAM (grup, pengguna, atau peran) (AWSAPI)

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 (API AWS)

Hubungi operasi IAM Access Analyzer berikut:

- [ValidatePolicy](#)

Memvalidasi 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 kendali akses IAM menggunakan AWS Management Console, AWS secara otomatis memeriksanya untuk memastikan mereka 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 memberikan pemeriksaan kebijakan tambahan dengan rekomendasi untuk membantu Anda menyempurnakan kebijakan lebih lanjut. Untuk mempelajari lebih lanjut 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 dihasilkan oleh IAM Access Analyzer, lihat [referensi pemeriksaan kebijakan IAM Access Analyzer](#).

Cakupan validasi

AWS memeriksa sintaks dan tata bahasa kebijakan JSON. Ini juga memverifikasi bahwa ARN Anda diformat dengan benar dan nama tindakan dan kunci syarat Anda benar.

Mengakses validasi kebijakan

Kebijakan divalidasi secara otomatis saat Anda membuat kebijakan JSON atau mengedit kebijakan yang ada diAWS Management Console. Jika sintaks kebijakan tidak valid, Anda menerima notifikasi dan harus memperbaiki masalahnya sebelum bisa melanjutkan. Temuan dari validasi kebijakan IAM Access Analyzer secara otomatis dihasilkan diAWS Management Console jika Anda memiliki izin untuk `access-analyzer:ValidatePolicy`. Anda juga dapat memvalidasi kebijakan menggunakan API AWS atauAWS 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 untuk memastikan izin yang aman dan fungsional. Sebaiknya buka kebijakan yang ada dan tinjau hasil validasi kebijakan yang dihasilkan. Anda tidak dapat mengedit dan menyimpan kebijakan yang sudah ada tanpa memperbaiki kesalahan sintaks kebijakan apa pun.

Hasilkan kebijakan berdasarkan aktivitas akses

Sebagai administrator atau developer, Anda dapat memberikan izin kepada entitas IAM (. IAM menyediakan beberapa opsi untuk membantu Anda menyempurnakan izin yang Anda berikan. Salah satu opsi adalah untuk menghasilkan kebijakan IAM yang didasarkan pada aktivitas akses untuk entitas. IAM Access Analyzer meninjauAWS CloudTrail log Anda dan menghasilkan templat kebijakan yang berisi izin yang entitas yang digunakan dalam rentang tanggal yang ditentukan. Anda

dapat menggunakan templat untuk membuat kebijakan dengan izin mendetail yang memberikan hanya izin yang diperlukan untuk mendukung kasus penggunaan khusus Anda.

Misalnya, bayangkan bahwa Anda adalah developer dan tim teknik Anda telah mengerjakan sebuah proyek untuk membuat aplikasi baru. Untuk mendorong eksperimen dan memungkinkan tim Anda bergerak cepat, Anda telah mengonfigurasi peran dengan izin yang luas saat aplikasi sedang dalam pengembangan. Sekarang aplikasi sudah siap untuk produksi. Sebelum aplikasi dapat memulai di akun produksi, Anda ingin mengidentifikasi hanya izin yang diperlukan peran agar aplikasi berfungsi. Ini membantu Anda untuk lebih menganut praktik terbaik [memberikan hak istimewa paling sedikit](#). Anda dapat membuat kebijakan berdasarkan aktivitas akses peran yang telah Anda gunakan untuk aplikasi di akun pengembangan. Anda dapat lebih menyempurnakan kebijakan yang dihasilkan dan kemudian melampirkan kebijakan ke entitas di akun produksi Anda.

Untuk mempelajari lebih lanjut tentang pembuatan kebijakan IAM Access Analyzer, lihat [pembuatan kebijakan IAM Access Analyzer](#).

Menguji kebijakan IAM dengan simulator kebijakan IAM

Untuk informasi lebih lanjut tentang bagaimana dan mengapa menggunakan kebijakan IAM, lihat [Kebijakan dan Izin di IAM](#).

[Anda dapat mengakses Konsol Simulator Kebijakan IAM di: https://policysim.aws.amazon.com/](https://policysim.aws.amazon.com/)

Important

Hasil simulator kebijakan dapat berbeda dari AWS lingkungan hidup Anda. Kami menyarankan Anda memeriksa kebijakan Anda terhadap AWS lingkungan hidup Anda setelah pengujian menggunakan simulator kebijakan untuk mengonfirmasi bahwa Anda memiliki hasil yang diinginkan. Untuk informasi selengkapnya, lihat [Bagaimana simulator kebijakan IAM bekerja](#).


[Memulai dengan IAM Policy Simulator](#)

Dengan simulator kebijakan IAM, Anda dapat menguji dan memecahkan masalah kebijakan berbasis identitas dan batas izin IAM. Berikut adalah beberapa hal umum yang bisa Anda lakukan dengan simulator kebijakan:

- Uji kebijakan berbasis identitas yang dilampirkan ke pengguna IAM, grup pengguna, atau peran dalam Anda. Akun AWS Jika lebih dari satu kebijakan terlampir ke pengguna, grup pengguna,


atau peran, Anda bisa menguji semua kebijakan, atau memilih kebijakan tertentu untuk diuji. Anda bisa menguji tindakan mana yang diizinkan atau ditolak oleh kebijakan terpilih untuk sumber daya tertentu.

- Menguji dan memecahkan masalah efek [batasan izin](#) pada entitas IAM. Anda hanya dapat mensimulasikan satu batas izin pada satu waktu.
- Uji efek kebijakan berbasis sumber daya pada pengguna IAM yang dilampirkan ke AWS sumber daya, seperti bucket Amazon S3, antrian Amazon SQS, topik Amazon SNS, atau kubah Amazon S3 Glacier. Untuk menggunakan kebijakan berbasis sumber daya dalam simulator kebijakan untuk pengguna IAM, Anda harus menyertakan sumber daya dalam simulasi. Anda juga harus memilih kotak centang untuk menyertakan kebijakan sumber daya tersebut dalam simulasi.

 Note

Simulasi kebijakan berbasis sumber daya tidak didukung untuk peran IAM.

- Jika Anda Akun AWS adalah anggota organisasi di [AWS Organizations](#), maka Anda dapat menguji dampak kebijakan kontrol layanan (SCP) pada kebijakan berbasis identitas Anda.

 Note

Simulator kebijakan tidak mengevaluasi SCP yang memiliki kondisi apa pun.

- Uji kebijakan berbasis identitas baru yang belum dilampirkan ke pengguna, grup pengguna, atau peran dengan mengetikkan atau menyalinnya ke dalam simulator kebijakan. Semua ini hanya digunakan dalam simulasi dan tidak disimpan. Anda tidak dapat mengetik atau menyalin kebijakan berbasis sumber daya di simulator kebijakan.
- Uji kebijakan berbasis identitas dengan layanan, tindakan, dan sumber daya yang dipilih. Misalnya, Anda bisa menguji untuk memastikan kebijakan Anda mengizinkan entitas untuk melakukan `ListAllMyBuckets`, `CreateBucket`, dan `DeleteBucket` tindakan di layanan Amazon S3 di bucket tertentu.
- Simulasikan skenario dunia nyata dengan menyediakan kunci konteks, misalnya alamat IP atau tanggal, yang disertakan ke `Condition` elemen dalam kebijakan yang diuji.

Note

Simulator kebijakan tidak mensimulasikan tag yang disediakan sebagai input jika kebijakan berbasis identitas dalam simulasi tidak memiliki `Condition` elemen yang secara eksplisit memeriksa tag.

- Identifikasi pernyataan spesifik mana dalam kebijakan berbasis identitas yang menghasilkan mengizinkan atau menolak akses ke sumber daya atau tindakan tertentu.

Topik

- [Bagaimana simulator kebijakan IAM bekerja](#)
- [Izin diperlukan untuk menggunakan simulator kebijakan IAM](#)
- [Menggunakan simulator kebijakan IAM \(konsol\)](#)
- [Menggunakan simulator kebijakan IAM \(AWS CLI dan AWS API\)](#)

Bagaimana simulator kebijakan IAM bekerja

Simulator kebijakan mengevaluasi pernyataan dalam kebijakan berbasis identitas dan masukan yang Anda berikan selama simulasi. Hasil simulator kebijakan dapat berbeda dari AWS lingkungan hidup Anda. Kami menyarankan Anda memeriksa kebijakan Anda terhadap AWS lingkungan hidup Anda setelah pengujian menggunakan simulator kebijakan untuk mengonfirmasi bahwa Anda memiliki hasil yang diinginkan.

Simulator kebijakan berbeda dari AWS lingkungan hidup dengan cara-cara berikut:

- Simulator kebijakan tidak membuat permintaan AWS layanan yang sebenarnya, sehingga Anda dapat menguji permintaan dengan aman yang mungkin membuat perubahan yang tidak diinginkan pada AWS lingkungan hidup Anda. Simulator kebijakan tidak mempertimbangkan nilai kunci konteks nyata dalam produksi.
- Karena simulator kebijakan tidak mensimulasikan menjalankan tindakan yang dipilih, simulator kebijakan tidak dapat melaporkan respons apa pun terhadap permintaan yang disimulasikan. Satu-satunya hasil yang dikembalikan adalah apakah tindakan yang diminta akan diizinkan atau ditolak.
- Jika Anda mengedit kebijakan di simulator kebijakan, perubahan ini hanya memengaruhi simulator kebijakan. Kebijakan terkait di Akun AWS tetap tidak berubah.
- Anda tidak dapat menguji kebijakan kontrol layanan (SCP) dengan kondisi apa pun.

- Simulator kebijakan tidak mendukung simulasi untuk peran IAM dan pengguna untuk akses lintas akun.

Note

Simulator kebijakan IAM tidak menentukan layanan mana yang mendukung [kunci kondisi global](#) untuk otorisasi. Misalnya, simulator kebijakan tidak mengidentifikasi bahwa layanan tidak mendukung [aws:TagKeys](#).

Izin diperlukan untuk menggunakan simulator kebijakan IAM

Anda bisa menggunakan konsol simulator kebijakan atau API simulator kebijakan untuk menguji kebijakan. Secara default, pengguna konsol dapat menguji kebijakan yang belum dilampirkan ke pengguna, grup pengguna, atau peran dengan mengetik atau menyalin kebijakan tersebut ke dalam simulator kebijakan. Kebijakan ini hanya dipakai dalam simulasi dan tidak mengungkapkan informasi sensitif. Pengguna API harus berizin untuk menguji kebijakan yang belum dilampirkan. Anda dapat mengizinkan pengguna konsol atau API untuk menguji kebijakan yang dilampirkan ke pengguna IAM, grup pengguna, atau peran dalam AndaAkun AWS. Untuk melakukannya, Anda harus mengizinkan pengambilan kebijakan tersebut. Guna menguji kebijakan berbasis sumber daya, pengguna harus berizin untuk mengambil kebijakan sumber daya tersebut.

Untuk contoh kebijakan konsol dan API yang memungkinkan pengguna menyimulasikan kebijakan, lihat [the section called “Contoh kebijakan: AWS Identity and Access Management \(IAM\)”](#).

Izin diperlukan untuk menggunakan konsol simulator kebijakan

Anda dapat mengizinkan pengguna untuk menguji kebijakan yang dilampirkan ke pengguna IAM, grup pengguna, atau peran dalam AndaAkun AWS. Untuk melakukannya, Anda harus mengizinkan pengguna Anda untuk mengambil kebijakan tersebut. Guna menguji kebijakan berbasis-sumber daya, pengguna harus berizin untuk mengambil kebijakan sumber daya tersebut.

Untuk melihat kebijakan contoh yang mengizinkan penggunaan konsol simulator kebijakan untuk kebijakan yang melekat ke pengguna, grup pengguna, atau peran, lihat [IAM:: Mengakses konsol simulator kebijakan](#).

Untuk melihat kebijakan contoh yang memungkinkan penggunaan konsol simulator kebijakan hanya untuk pengguna dengan jalur tertentu, lihat [IAM: Mengakses konsol simulator kebijakan berdasarkan jalur](#).

Untuk membuat kebijakan yang memungkinkan penggunaan konsol simulator kebijakan hanya untuk satu tipe entitas, gunakan prosedur berikut.

Untuk memungkinkan pengguna konsol mensimulasikan kebijakan bagi pengguna

Sertakan tindakan berikut dalam kebijakan Anda:

- `iam:GetGroupPolicy`
- `iam:GetPolicy`
- `iam:GetPolicyVersion`
- `iam:GetUser`
- `iam:GetUserPolicy`
- `iam>ListAttachedUserPolicies`
- `iam>ListGroupsForUser`
- `iam>ListGroupPolicies`
- `iam>ListUserPolicies`
- `iam>ListUsers`

Untuk memungkinkan pengguna konsol mensimulasikan kebijakan untuk grup pengguna

Sertakan tindakan berikut dalam kebijakan Anda:

- `iam:GetGroup`
- `iam:GetGroupPolicy`
- `iam:GetPolicy`
- `iam:GetPolicyVersion`
- `iam>ListAttachedGroupPolicies`
- `iam>ListGroupPolicies`
- `iam>ListGroups`

Untuk memungkinkan pengguna konsol mensimulasikan kebijakan untuk peran

Sertakan tindakan berikut dalam kebijakan Anda:

- `iam:GetPolicy`

- `iam:GetPolicyVersion`
- `iam:GetRole`
- `iam:GetRolePolicy`
- `iam>ListAttachedRolePolicies`
- `iam>ListRolePolicies`
- `iam>ListRoles`

Untuk menguji kebijakan berbasis-sumber daya, pengguna harus berizin untuk mengambil kebijakan sumber daya tersebut.

Untuk memungkinkan pengguna konsol menguji kebijakan berbasis sumber daya di bucket Amazon S3

Sertakan tindakan berikut dalam kebijakan Anda:

- `s3:GetBucketPolicy`

Misalnya, kebijakan berikut memakai tindakan ini untuk memungkinkan pengguna konsol mensimulasikan kebijakan berbasis sumber daya dalam bucket Amazon S3 tertentu.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:GetBucketPolicy",
      "Resource": "arn:aws:s3:::bucket-name/*"
    }
  ]
}
```

Izin diperlukan untuk menggunakan API simulator kebijakan

Operasi API simulator kebijakan [GetContextKeyForCustomPolicy](#) dan [SimulateCustomPolicy](#) memungkinkan Anda menguji kebijakan yang belum dilampirkan ke pengguna, grup pengguna, atau peran. Untuk menguji kebijakan tersebut, Anda meneruskan kebijakan sebagai string ke API. Kebijakan ini hanya dipakai dalam simulasi dan tidak mengungkap informasi sensitif. Anda juga dapat menggunakan API untuk menguji kebijakan yang dilampirkan ke pengguna IAM,

grup pengguna, atau peran dalam AndaAkun AWS. Untuk melakukan itu, Anda harus memberi pengguna izin untuk menelepon [GetContextKeyForPrincipalPolicy](#) dan [SimulatePrincipalPolicy](#).

Untuk melihat contoh kebijakan yang memungkinkan penggunaan API simulator kebijakan untuk kebijakan terlampir dan tidak terikat saat iniAkun AWS, lihat [IAM: Mengakses API simulator kebijakan: Mengakses API simulator kebijakan Mengakses API simulator kebijakan](#).

Untuk membuat kebijakan yang memungkinkan penggunaan konsol API simulator kebijakan hanya untuk satu tipe kebijakan, gunakan prosedur berikut.

Untuk memungkinkan pengguna API menyimulasikan kebijakan yang diteruskan secara langsung ke API sebagai string

Sertakan tindakan berikut dalam kebijakan Anda:

- `iam:GetContextKeysForCustomPolicy`
- `iam:SimulateCustomPolicy`

Untuk memungkinkan pengguna API mensimulasikan kebijakan yang dilampirkan ke pengguna IAM, grup pengguna, peran, atau sumber daya

Sertakan tindakan berikut dalam kebijakan Anda:

- `iam:GetContextKeysForPrincipalPolicy`
- `iam:SimulatePrincipalPolicy`

Misalnya, untuk memberi pengguna bernama Bob izin menyimulasikan kebijakan yang ditetapkan untuk pengguna bernama Alice, beri Bob akses ke sumber daya berikut ini:
`arn:aws:iam::777788889999:user/alice`.

Untuk melihat kebijakan contoh yang memungkinkan penggunaan API simulator kebijakan hanya untuk pengguna dengan jalur tertentu, lihat [IAM: Mengakses API simulator kebijakan berdasarkan jalur pengguna](#).

Menggunakan simulator kebijakan IAM (konsol)

Secara default, pengguna konsol dapat menguji kebijakan yang belum dilampirkan ke pengguna, grup pengguna, atau peran dengan mengetik atau menyalin kebijakan tersebut ke konsol simulator kebijakan. Kebijakan ini hanya dipakai dalam simulasi dan tidak mengungkap informasi sensitif.

Untuk menguji kebijakan yang tidak melekat ke pengguna, grup pengguna, atau peran (konsole)

1. Buka konsol simulator kebijakan IAM di: <https://policysim.aws.amazon.com/>
2. Di menuMode: pada bagian atas halaman, pilih Kebijakan Baru.
3. Di Policy Sandbox, pilih Buat Kebijakan Baru.
4. Ketik atau salin kebijakan ke dalam simulator kebijakan, dan gunakan simulator kebijakan seperti yang dijelaskan dalam langkah-langkah berikut.

Setelah memiliki izin untuk menggunakan Konsol Simulator Kebijakan IAM, Anda dapat menggunakan simulator kebijakan untuk menguji kebijakan pengguna, grup pengguna, peran, atau sumber daya IAM.

Untuk menguji kebijakan yang melekat ke pengguna, grup pengguna, atau peran (konsole)

1. Buka konsol simulator kebijakan IAM di: <https://policysim.aws.amazon.com/>

Note

Untuk masuk ke simulator kebijakan sebagai pengguna IAM, gunakan URL login unik Anda untuk masuk ke AWS Management Console Lalu pergi ke <https://policysim.aws.amazon.com/>. Untuk informasi lebih lanjut mengenai masuk sebagai pengguna IAM, lihat [Cara pengguna IAM M M M IAM IAM IAM IAM IMAAWS](#).

Simulator kebijakan terbuka dalam mode Kebijakan yang Ada dan mencantumkan pengguna IAM di akun Anda di bawah Pengguna, Grup, dan Peran.

2. Pilih opsi yang sesuai dengan tugas Anda:

Untuk menguji ini:	Lakukan ini:
Kebijakan yang melekat pada pengguna	Pilih Pengguna pada daftarPengguna, Grup, dan Peran. Lalu pilih pengguna.
Kebijakan yang melekat pada grup pengguna	Pilih Grup pada daftarPengguna, Grup, dan Peran. Lalu pilih grup pengguna.

Untuk menguji ini:	Lakukan ini:
Kebijakan yang melekat pada peran	Pilih Peran pada daftar Pengguna, Grup, dan Peran. Lalu pilih peran.
Kebijakan yang terlampir pada sumber daya	Lihat Step 9 .
Kebijakan khusus untuk pengguna, grup pengguna, atau peran	Pilih Buat Kebijakan Baru. Pada panel Kebijakan baru, ketik atau tempelkan kebijakan dan pilih Terapkan.

Kiat


Untuk menguji kebijakan yang dilampirkan ke grup pengguna, Anda dapat meluncurkan simulator kebijakan IAM langsung dari [konsol IAM](#): Di panel navigasi, pilih Grup pengguna. Pilih nama grup yang dimana Anda ingin menguji kebijakan, dan kemudian pilih tab Izin. Pilih Simulasi.

Untuk menguji kebijakan yang dikelola pelanggan yang terlampir pada pengguna: Di panel navigasi, pilih Pengguna. Pilih nama pengguna dengan siapa Anda ingin menguji kebijakan. Lalu pilih tab Izin dan perluas kebijakan yang ingin Anda uji. Di ujung kanan, pilih Simulasikan kebijakan. Simulator Kebijakan IAM terbuka di jendela baru dan menampilkan kebijakan terpilih di panel Kebijakan.

- (Opsional) Jika akun Anda adalah anggota sebuah organisasi di [AWS Organizations](#), pilih kotak centang di samping AWS Organizations SCP untuk memasukkan SCP evaluasi Anda yang disimulasikan. SCP adalah kebijakan JSON yang menentukan izin maksimum untuk sebuah organisasi atau unit organisasional (OU). SCP membatasi izin untuk entitas di akun anggota. Jika SCP memblokir layanan atau tindakan, maka tidak ada entitas pada akun tersebut yang bisa mengakses layanan itu atau serta tidak bisa melakukan tindakan tersebut. Hal ini juga berlaku bahkan jika administrator dengan tegas mengizinkan layanan atau tindakan tersebut melalui IAM atau kebijakan sumber daya.

Jika akun Anda bukan anggota sebuah organisasi, maka kotak centang tidak akan muncul.

4. (Opsional) Anda dapat menguji kebijakan yang ditetapkan sebagai [batas izin](#) untuk entitas IAM (pengguna atau peran), tetapi tidak untuk grup pengguna. Jika kebijakan batasan izin saat ini diatur untuk entitas, ia muncul dalam panel Kebijakan. Anda hanya bisa mengatur satu batasan izin untuk sebuah entitas. Untuk menguji batasan izin yang berbeda, Anda bisa membuat batasan izin khusus. Untuk melakukannya, pilih Buat Kebijakan Baru. Panel Kebijakan baru akan terbuka. Di menu, pilih Kebijakan Batasan Izin IAM Khusus. Masukkan nama untuk kebijakan baru dan ketik atau salin kebijakan ke ruang di bawah ini. Pilih Terapkan untuk menyimpan kebijakan tersebut. Selanjutnya, pilih Kembali untuk kembali ke panel Kebijakan awal. Lalu pilih kotak centang di samping batasan izin yang ingin Anda pakai untuk simulasi.
5. (Opsional) Anda hanya bisa menguji subset kebijakan yang melekat pada pengguna, grup pengguna, atau peran. Untuk melakukannya, di panel Kebijakan kosongkan kotak centang di samping setiap kebijakan ingin Anda keluarkan.
6. Di bawah Simulator Kebijakan, pilih Pilih layanan lalu pilih layanan yang ingin diuji. Lalu pilih Pilih tindakan dan pilih satu atau lebih tindakan untuk diuji. Walaupun menu menunjukkan pilihan yang tersedia hanya untuk satu layanan pada satu waktu, semua layanan dan tindakan yang telah Anda pilih muncul di Pengaturan Tindakan dan Hasil.
7. (Opsional) Jika salah satu kebijakan yang Anda pilih [Step 2](#) dan [Step 5](#) menyertakan kondisi dengan [kunci kondisi AWS global](#), berikan nilai untuk kunci tersebut. Anda melakukan hal ini dengan memperluas bagian Pengaturan Global dan mengetikkan nilai untuk nama utama yang ditampilkan di sana.

 Warning

Jika Anda membiarkan nilai untuk kunci kondisi kosong, maka kunci tersebut akan diabaikan selama simulasi. Pada beberapa kasus, hal ini menghasilkan kesalahan, dan simulasi gagal untuk dijalankan. Pada kasus lain, simulasi berjalan, namun hasilnya mungkin tidak bisa diandalkan. Pada kasus tersebut, simulasi tidak sesuai dengan kondisi dunia nyata yang mencakup nilai untuk kunci kondisi atau variabel.

8. (Opsional) Setiap tindakan terpilih muncul di daftar Pengaturan Tindakan dan Hasil dengan Tidak disimulasikan muncul di kolom Izin sampai Anda benar-benar menjalankan simulasi tersebut. Sebelum menjalankan simulasi, Anda bisa mengonfigurasi setiap tindakan dengan sumber daya. Untuk mengonfigurasi tindakan individu untuk skenario tertentu, pilih panah untuk memperluas baris tindakan. Bila tindakan mendukung izin tingkat-sumber daya, Anda bisa mengetik [Amazon Resource Name \(ARN\)](#) dari sumber daya tertentu yang aksesnya ingin Anda uji. Secara default, setiap sumber daya diatur sebagai wildcard (*). Anda juga bisa menentukan nilai tertentu

untuk [kunci konteks kondisi mana pun](#). Seperti disebut sebelumnya, kunci dengan nilai kosong diabaikan, yang bisa menimbulkan kegagalan simulasi atau hasil yang tidak bisa diandalkan.

- a. Pilih panah di samping nama tindakan untuk memperluas setiap baris dan mengonfigurasi tambahan informasi yang diperlukan untuk secara akurat menyimulasikan tindakan pada skenario Anda. Bila tindakan tersebut membutuhkan izin tingkat-sumber daya, Anda bisa mengetik [Amazon Resource Name \(ARN\)](#) dari sumber daya tertentu yang Anda ingin simulasikan aksesnya. Secara default, setiap sumber daya diatur sebagai wildcard (*).
- b. Bila tindakan tersebut mendukung izin tingkat sumber daya namun tidak memerlukannya, maka Anda bisa memilih Tambah Sumber Daya untuk memilih tipe sumber daya yang ingin Anda tambahkan ke simulasi.
- c. Jika salah satu kebijakan terpilih mencakup Condition elemen yang merujuk ke kunci konteks untuk layanan tindakan ini, maka nama kunci tersebut akan muncul di bawah tindakan. Anda dapat menetapkan nilai yang akan dipakai selama simulasi tindakan tersebut untuk sumber daya yang ditentukan.

Tindakan yang memerlukan grup tipe sumber daya berbeda

Beberapa tindakan memerlukan tipe sumber daya yang berbeda dengan keadaan berbeda. Setiap grup tipe sumber daya terkait dengan sebuah skenario. Jika salah satu dari ini berlaku untuk simulasi Anda, pilih dan simulator kebijakan memerlukan jenis sumber daya yang sesuai untuk skenario itu. Daftar berikut menunjukkan setiap opsi skenario yang didukung dan sumber daya yang harus Anda tentukan untuk menjalankan simulasi tersebut.

Masing-masing skenario Amazon EC2 berikut ini mengharuskan Anda untuk menentukan instance, image, dan security-group sumber daya. Jika skenario Anda mencakup volume EBS, maka Anda harus menjelaskannya volume sebagai sumber daya. Jika skenario Amazon EC2 mencakup Virtual Private Cloud (VPC) maka Anda harus memasok network-interface sumber dayanya. Jika termasuk IP subnet, maka Anda harus menentukan subnet sumber dayanya. Untuk informasi lebih lanjut tentang opsi skenario Amazon EC2, lihat [Platform yang Didukung](#) di Panduan Pengguna Amazon EC2 .

- EC2-VPC InstanceStore
instance, image, security-group, network-interface
- EC2-VPC-Subnet InstanceStore

instance, image, security-group, network-interface, subnet

- EC2-VPC-EBS

instance, image, security-group, network-interface, volume

- EC2-VPC-EBS-Subnet

instance, image, security-group, network-interface, subnet, volume

9. (Opsional) Jika Anda ingin memasukkan kebijakan berbasis sumber daya dalam simulasi Anda, maka terlebih dahulu Anda harus memilih tindakan yang ingin Anda simulasikan pada sumber daya tersebut dalam [Step 6](#). Perluas baris untuk tindakan terpilih, dan ketik ARN sumber daya tersebut dengan kebijakan yang ingin Anda simulasikan. Lalu pilih Sertakan Kebijakan Sumber Daya di samping kotak teks ARN. Simulator kebijakan IAM saat ini mendukung kebijakan berbasis sumber daya hanya dari layanan berikut: Amazon S3 (hanya kebijakan berbasis sumber daya; ACL saat ini tidak didukung), Amazon SQS, Amazon SNS, dan brankas S3 Glacier yang tidak terkunci (brankas terkunci saat ini tidak didukung).
10. Pilih Jalankan Simulasi di sudut kanan atas.

Izin kolom di setiap baris Pengaturan Tindakan dan Hasil menampilkan hasil simulasi dari tindakan tersebut di sumber daya yang ditentukan.

11. Untuk melihat pernyataan mana dalam sebuah kebijakan yang mengizinkan atau menolak tindakan, pilih tautan **N** Pernyataan yang cocok di kolom Izin untuk memperluas baris. Lalu pilih tautan Tampilkan pernyataan. Panel Kebijakan menunjukkan kebijakan yang relevan dengan pernyataan yang memengaruhi hasil simulasi yang disoroti.

Note

Jika sebuah tindakan secara implisit ditolak—yaitu, jika tindakan tersebut ditolak hanya karena tidak secara eksplisit diizinkan—opsi Daftar dan Tampilkan pernyataan tidak ditampilkan.

Pemecahan masalah pesan konsol simulator kebijakan IAM

Tabel berikut mencantumkan pesan informasi dan peringatan yang mungkin Anda jumpai saat menggunakan simulator kebijakan IAM. Tabel ini juga memberikan langkah-langkah yang bisa Anda ambil untuk menyelesaikannya.

Pesan	Langkah-langkah untuk menyelesaikan
<p>Kebijakan ini telah diedit Perubahan tidak akan disimpan ke akun Anda.</p>	<p>Tidak ada tindakan yang diperlukan.</p> <p>Pesan ini bersifat informatif Jika Anda mengedit kebijakan yang ada di simulator kebijakan IAM, perubahan Anda tidak akan memengaruhi kebijakan AndaAkun AWS. Simulator kebijakan memungkinkan Anda membuat perubahan pada kebijakan hanya untuk tujuan pengujian.</p>
<p>Tidak bisa mengambil kebijakan sumber daya. Alasan: <i>Pesan kesalahan mendetail</i></p>	<p>Simulator kebijakan tidak dapat mengakses kebijakan berbasis sumber daya yang diminta. Pastikan ARN sumber daya yang ditentukan sudah benar dan pengguna yang menjalankan simulasi memiliki izin untuk membaca kebijakan sumber daya.</p>
<p>Satu atau lebih kebijakan memerlukan nilai dalam pengaturan simulasi. Simulasi bisa gagal tanpa nilai ini.</p>	<p>Pesan ini muncul jika kebijakan yang sedang Anda uji berisi kunci kondisi atau variabel namun Anda tidak memberi nilai apa pun untuk kunci atau variabel tersebut diPengaturan Simulasi.</p> <p>Untuk mengabaikan pesan ini, pilih Pengaturan Simulasi, Lalu masukkan nilai untuk setiap kunci kondisi atau variabel.</p>
<p>Anda telah mengubah kebijakan. Hasil ini tidak lagi valid.</p>	<p>Pesan ini muncul jika Anda telah mengubah kebijakan yang dipilih ketika hasil ditampilkan di panelHasil. Hasil yang ditampilkan di panel Hasil tidak diperbarui secara dinamis.</p> <p>Untuk mengabaikan pesan ini, pilih Jalankan Simulasi lagi untuk menampilkan hasil simulasi baru berdasarkan perubahan yang dibuat di panelKebijakan.</p>

Pesan	Langkah-langkah untuk menyelesaikan
<p>Sumber daya yang Anda ketikkan untuk simulasi ini tidak cocok dengan layanan ini.</p>	<p>Pesan ini muncul jika Anda sudah mengetikkan Amazon Resource Name (ARN) di panel Pengaturan Simulasi yang tidak cocok dengan layanan yang Anda pilih untuk simulasi saat ini. Misalnya, pesan ini muncul jika Anda menentukan ARN untuk sumber daya Amazon DynamoDB tetapi Anda memilih Amazon Redshift sebagai layanan yang akan disimulasikan.</p> <p>Untuk mengabaikan pesan ini, lakukan salah satu hal berikut:</p> <ul style="list-style-type: none">• Lepaskan ARN dari kotak pada panel Pengaturan Simulasi.• Pilih layanan yang cocok dengan ARN yang Anda tentukan dalam Pengaturan Simulasi.
<p>Tindakan ini termasuk dalam layanan yang mendukung mekanisme kontrol akses khusus selain kebijakan berbasis sumber daya, seperti kebijakan Amazon S3 ACL atau S3 Glacier vault lock. Simulator kebijakan tidak mendukung mekanisme ini, sehingga hasilnya bisa berbeda dari lingkungan produksi Anda.</p>	<p>Tidak ada tindakan yang diperlukan.</p> <p>Pesan ini bersifat informatif. Dalam versi saat ini, simulator kebijakan mengevaluasi kebijakan yang dilampirkan ke pengguna dan grup pengguna, serta dapat mengevaluasi kebijakan berbasis sumber daya untuk Amazon S3, Amazon SQS, Amazon SNS, dan S3 Glacier. Simulator kebijakan tidak mendukung semua mekanisme kendali akses yang didukung oleh layanan AWS lainnya.</p>

Pesan	Langkah-langkah untuk menyelesaikan
<p>DynamoDB FGAC saat ini tidak didukung.</p>	<p>Tidak ada tindakan yang diperlukan.</p> <p>Pesan informatif ini merujuk ke kendali akses fine-grained. Kontrol akses berbutir halus adalah kemampuan untuk menggunakan kondisi kebijakan IAM untuk menentukan siapa yang dapat mengakses item dan atribut data individual dalam tabel dan indeks DynamoDB. Juga merujuk ke tindakan yang bisa dilakukan pada tabel dan indeks ini. Versi simulator terkini dari simulator kebijakan IAM tidak mendukung tipe kondisi kebijakan semacam ini. Untuk informasi selengkapnya tentang kontrol akses berbutir halus DynamoDB, lihat Kontrol Akses Berbutir Baik untuk DynamoDB.</p>
<p>Anda memiliki kebijakan yang tidak mematuhi sintaksis kebijakan. Anda dapat menggunakan validasi kebijakan untuk meninjau pembaruan yang disarankan bagi kebijakan Anda.</p>	<p>Pesan ini muncul di bagian atas daftar kebijakan jika Anda memiliki kebijakan yang tidak mematuhi tata bahasa kebijakan IAM. Untuk menyimulasikan kebijakan ini, tinjau opsi validasi kebijakan di Memvalidasi kebijakan IAM untuk mengidentifikasi dan memperbaiki kebijakan ini.</p>
<p>Kebijakan ini harus diperbarui agar mematuhi aturan sintaksis kebijakan terbaru.</p>	<p>Pesan ini muncul jika Anda memiliki kebijakan yang tidak mematuhi tata bahasa kebijakan IAM. Untuk menyimulasikan kebijakan ini, tinjau opsi validasi kebijakan di Memvalidasi kebijakan IAM untuk mengidentifikasi dan memperbaiki kebijakan ini.</p>

Menggunakan simulator kebijakan IAM (AWS CLI dan AWS API)

Perintah simulator kebijakan biasanya butuh memanggil operasi API untuk melakukan dua hal:

1. Mengevaluasi kebijakan dan mengembalikan daftar kunci konteks yang mereka referensikan. Anda perlu tahu kunci konteks mana yang direferensikan sehingga Anda bisa memasok nilai bagi mereka di langkah berikutnya.
2. Simulasikan kebijakan, berikan daftar tindakan, sumber daya, dan kunci konteks yang digunakan selama simulasi.

Untuk alasan keamanan, operasi API telah dibagi menjadi dua grup:

- Operasi API yang menyimulasikan hanya kebijakan yang diteruskan secara langsung ke API sebagai string. Set ini termasuk [GetContextKeysForCustomPolicy](#) dan [SimulateCustomPolicy](#).
- Operasi API yang mensimulasikan kebijakan yang dilampirkan ke pengguna, grup pengguna, peran, atau sumber daya IAM tertentu. Karena operasi API ini dapat mengungkapkan rincian izin yang diberikan ke entitas IAM lainnya, Anda harus mempertimbangkan pembatasan akses ke operasi API ini. Set ini termasuk [GetContextKeysForPrincipalPolicy](#) dan [SimulatePrincipalPolicy](#). Untuk informasi lebih lanjut tentang pembatasan akses ke operasi API, lihat [Contoh kebijakan: AWS Identity and Access Management \(IAM\)](#).

Di kedua kasus tersebut, operasi API menyimulasikan efek dari satu atau lebih kebijakan dari daftar tindakan dan sumber daya. Setiap tindakan dipasangkan dengan setiap sumber daya dan simulasi menentukan apakah kebijakan itu mengizinkan atau menolak tindakan untuk sumber daya tersebut. Anda juga bisa memberi nilai bagi setiap kunci konteks yang menjadi referensi kebijakan Anda. Anda bisa mendapatkan daftar kunci konteks yang referensi kebijakan dengan terlebih dahulu memanggil [GetContextKeysForCustomPolicy](#) atau [GetContextKeysForPrincipalPolicy](#). Jika Anda tidak memberikan nilai untuk kunci konteks, simulasi masih berjalan. Tetapi hasilnya mungkin tidak dapat diandalkan karena simulator kebijakan tidak dapat memasukkan kunci konteks itu dalam evaluasi.

Untuk mendapatkan daftar kunci konteks (AWS CLI, API AWS)

Gunakan hal berikut untuk mengevaluasi daftar kebijakan dan mengembalikan daftar kunci konteks yang dipakai dalam kebijakan.

- AWS CLI: [aws iam get-context-keys-for-custom-policy](#) dan [aws iam get-context-keys-for-principal-policy](#)
- API AWS: [GetContextKeysForCustomPolicy](#) dan [GetContextKeysForPrincipalPolicy](#)

Untuk mensimulasikan kebijakan IAM (AWS CLI, AWS API)

Gunakan hal berikut untuk mensimulasikan kebijakan IAM untuk menentukan izin efektif pengguna.

- AWS CLI: [aws iam simulate-custom-policy](#) dan [aws iam simulate-principal-policy](#)
- API AWS: [SimulateCustomPolicy](#) dan [SimulatePrincipalPolicy](#)

Menambahkan dan menghapus izin identitas IAM

Anda menggunakan kebijakan untuk menentukan izin bagi identitas (pengguna, grup pengguna, atau peran). Anda bisa menambah dan menghapus izin dengan melampirkan dan memisahkan kebijakan IAM untuk identitas dengan AWS Command Line Interface menggunakan, atau AWS API. AWS Management Console AWS CLI Anda juga bisa memakai kebijakan untuk menetapkan [batasan izin](#) hanya untuk entitas (pengguna atau peran) yang menggunakan metode yang sama. Batasan izin adalah fitur lanjutan AWS yang mengendalikan izin maksimum yang bisa dimiliki oleh sebuah entitas.

Topik


- [Terminologi](#)
- [Lihat aktivitas identitas](#)
- [Menambahkan izin identitas IAM \(konsol\)](#)
- [Menghapus izin identitas IAM \(konsol\)](#)
- [Menambahkan kebijakan AWS CLI](#)
- [Menghapus kebijakan AWS CLI](#)
- [Menambahkan kebijakan \(AWS API\)](#)
- [Menghapus kebijakan \(AWS API\)](#)

Terminologi

Ketika Anda mengasosiasikan izin kebijakan dengan identitas (pengguna, grup pengguna, dan peran), terminologi dan prosedur bervariasi tergantung apakah Anda bekerja dengan kebijakan terkelola atau inline:


- 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 melepas 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 inline dari identitas IAM (pengguna, grup pengguna, atau peran). Menghapus kebijakan akan menghapus izinnya dari identitas.

 Note

Anda bisa menghapus kebijakan inline untuk [peran terkait layanan](#) hanya dalam layanan yang bergantung pada peran tersebut. Lihat [dokumentasi AWS](#) dari layanan Anda untuk melihat apakah ia mendukung fitur ini.

Anda bisa memakai konsol, AWS CLI, atau API AWS 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 IAM](#).
- Untuk informasi tentang memvalidasi kebijakan IAM, lihat [Memvalidasi kebijakan IAM](#).
- Jumlah dan ukuran sumber daya IAM diAWS 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 [Menyaring izin di AWS menggunakan informasi yang terakhir diakses](#).

Menambahkan izin identitas IAM (konsol)

Anda bisa memakai AWS Management Console untuk menambah 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 AWS Management Console 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 samping nama kebijakan yang 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 AWS Management Console 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 rincian kebijakan, pilih tab Entitas, lalu, bila perlu, buka bagian Terlampir sebagai batasan izin dan pilih Tetapkan kebijakan ini sebagai batasan 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 prinsipiel, pilih Tetapkan batasan izin

Untuk menyematkan kebijakan inline bagi pengguna atau peran (konsole)

1. Masuk ke AWS Management Console 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 pada [peran 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 yang dikelola bisa menjadi kebijakan yang dikelola AWS atau kebijakan yang dikelola pelanggan yang sebelumnya Anda buat.
 - [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](#)— Pada opsi editor JSON, Anda bisa memakai sintaksis JSON untuk membuat kebijakan. Anda bisa mengetik dokumen kebijakan JSON yang baru atau menempelkan satu [contoh 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 AWS Management Console 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 dari 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 AWS Management Console 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 rincian kebijakan, pilih tab Entitas, lalu, bila perlu, buka bagian Terlampir sebagai batasan izin Centang kotak di samping pengguna atau peran yang batasannya 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 Tetapkan batasan izin

Menghapus izin identitas IAM (konsol)

Anda bisa memakai AWS Management Console 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 AWS Management Console 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 samping nama kebijakan yang ingin 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 AWS Management Console 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, lalu, bila perlu, buka bagian Terlampir sebagai batasan izin dan pilih entitas yang ingin menghapus batasan izin. Lalu pilih Hapus batas.
5. Konfirmasi bahwa Anda ingin menghapus batasan dan pilih Hapus batasan.

Untuk menghapus kebijakan inline (konsol)

1. Masuk ke AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Di panel navigasi, pilih Grup pengguna, Pengguna, atau Peran.

3. Dari daftar, pilih nama grup pengguna, pengguna, atau peran yang memiliki kebijakan yang ingin Anda hapus.
4. Pilih tab Izin.
5. Centang kotak di samping kebijakan dan pilih Hapus.
6. Pilih Hapus di kotak konfirmasi.

Menambahkan kebijakan AWS CLI

Anda bisa memakai AWS CLI untuk menambah 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 attach-user-policy](#)
 - [AWS attach-group-policy](#)
 - [AWS 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 put-user-permissions-boundary](#)

- [AWS 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 put-user-policy](#)
- [AWS put-group-policy](#)
- [AWS put-role-policy](#)

Menghapus kebijakan AWS CLI

Anda bisa memakai AWS CLI untuk melepas 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 (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, grup pengguna, dan peran) yang telah terlampir dengan kebijakan terkelola:
 - [AWS list-entities-for-policy](#)
 - Untuk mendaftar kebijakan terkelola yang menempel pada identitas (pengguna, grup pengguna, atau peran), gunakan salah satu perintah berikut:
 - [AWS list-attached-user-policies](#)
 - [AWS list-attached-group-policies](#)
 - [AWS list-attached-role-policies](#)
3. Untuk melepas kebijakan terkelola dari identitas (pengguna, grup pengguna, atau peran), gunakan salah satu perintah berikut:

- [AWS detach-user-policy](#)
- [AWS detach-group-policy](#)
- [AWS 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](#)
 - [aws](#)
2. (Opsional) Untuk melihat pengguna atau peran dimana kebijakan terkelola dipakai untuk batasan izin, jalankan perintah berikut:
 - [AWS 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 delete-user-permissions-boundary](#)
 - [AWS delete-role-permissions-boundary](#)

Untuk menghapus kebijakan inline (AWS CLI)

1. (Opsional) Untuk mendaftarkan seluruh kebijakan inline yang dilampirkan pada identitas (pengguna, grup pengguna, atau peran), gunakan salah satu perintah berikut:
 - [AWS list-user-policies](#)
 - [AWS list-group-policies](#)
 - [AWS 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 get-user-policy](#)

- [AWS get-group-policy](#)
 - [AWS 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 delete-user-policy](#)
 - [AWS delete-group-policy](#)
 - [AWS delete-role-policy](#)

Menambahkan kebijakan (AWSAPI)

Anda dapat menggunakan API AWS untuk melampirkan kebijakan terkelola yang mengontrol izin atau menentukan kebijakan yang berfungsi sebagai [batasan perizinan](#). Anda juga bisa menyematkan kebijakan inline.

Untuk menggunakan kebijakan terkelola sebagai kebijakan perizinan untuk suatu entitas (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. Untuk melampirkan kebijakan terkelola pada sebuah identitas (pengguna, grup pengguna, atau peran), hubungi salah satu layanan berikut:
 - [AttachUserPolicy](#)
 - [AttachGroupPolicy](#)
 - [AttachRolePolicy](#)

Untuk menggunakan kebijakan terkelola pada penetapan batasan perizinan (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 menyimpan 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 (AWSAPI)

Anda dapat menggunakan API AWS untuk memisahkan kebijakan terkelola yang mengontrol izin atau menghapus kebijakan yang berfungsi sebagai batasan [batasan perizinan](#). Anda juga bisa menghapus kebijakan inline.

Untuk melepaskan kebijakan terkelola yang digunakan sebagai kebijakan perizinan (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, grup pengguna, dan peran) yang telah terlampir dengan 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 perizinan (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 (API AWS)

1. (Opsional) Untuk mendaftar seluruh kebijakan inline yang dilampirkan pada identitas (pengguna, grup pengguna, atau peran), hubungi salah satu layanan berikut:
 - [ListUserPolicies](#)
 - [ListGroupPolicies](#)
 - [ListRolePolicies](#)

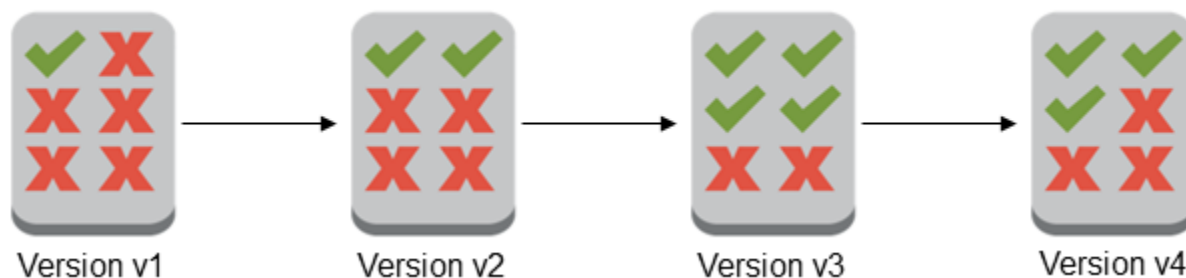
2. (Opsional) Untuk memperoleh dokumen kebijakan inline yang disematkan di identitas (pengguna, grup pengguna, atau peran), hubungi salah satu layanan berikut:
 - [GetUserPolicy](#)
 - [GetGroupPolicy](#)
 - [GetRolePolicy](#)
3. Untuk menghapus kebijakan inline dari suatu identitas (pengguna, grup pengguna, 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.

Multiple versions of a single managed policy



Versi kebijakan berbeda dengan `Version` elemen kebijakan. Elemen kebijakan `Version` digunakan dalam kebijakan dan menentukan versi bahasa kebijakan. Untuk mempelajari lebih lanjut tentang `Version` elemen kebijakan lihat [Elemen 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

- [Izin untuk mengatur versi default kebijakan](#)
- [Mengatur versi default kebijakan terkelola pelanggan](#)
- [Menggunakan beberapa versi untuk mengembalikan perubahan](#)
- [Pembatasan versi](#)

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

Menolak tindakan `iam:SetDefaultPolicyVersion` dalam kebijakan pengguna tidak menghentikan pengguna untuk membuat versi kebijakan baru dan mengaturnya sebagai default.

Anda dapat menggunakan kebijakan berikut untuk menolak akses pengguna untuk mengubah kebijakan terkelola pelanggan yang sudah ada:

```
{
  "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

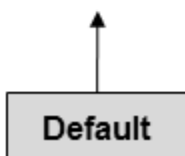
Salah satu versi kebijakan terkelola ditetapkan sebagai versi default. Versi default kebijakan adalah versi operatif—yaitu, ini adalah versi yang berlaku untuk semua entitas utama (pengguna, grup pengguna, dan peran) tempat kebijakan terkelola dilampirkan.

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 terkelola AWS, versi default diatur oleh AWS. Diagram berikut mengilustrasikan hal ini.

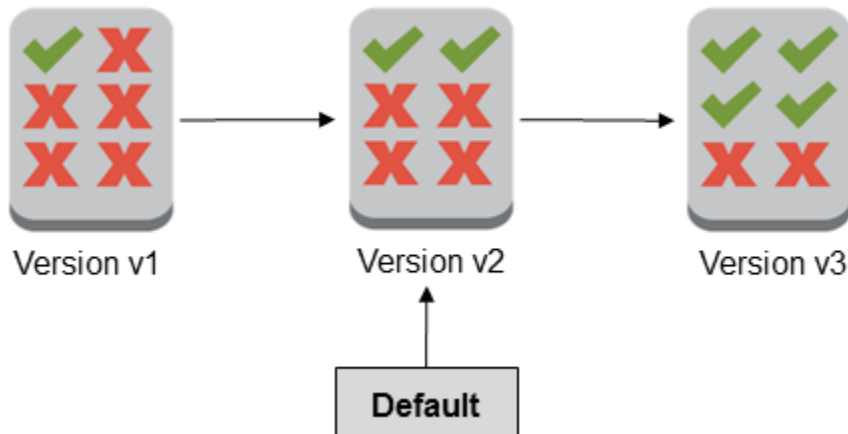
Managed policy with one version



Version v1



Managed policy with multiple versions



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 menetapkan versi default untuk kebijakan terkelola AWS atau kebijakan inline.

Untuk mengatur versi default kebijakan terkelola pelanggan (konsol)

1. Masuk ke AWS Management Console 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.

Untuk mempelajari cara mengatur versi default kebijakan terkelola pelanggan dari AWS Command Line Interface atau API AWS, lihat [Mengedit kebijakan terkelola pelanggan \(AWS CLI\)](#).

Menggunakan beberapa versi untuk mengembalikan 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 AWS Management Console. 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.

Pembatasan versi

Kebijakan terkelola dapat memiliki sampai dengan lima versi. Apabila Anda perlu melakukan perubahan pada kebijakan terkelola di luar lima versi dari AWS Command Line Interface, atau AWS API, Anda harus menghapus satu atau lebih versi yang ada terlebih dahulu. Apabila Anda menggunakan AWS Management Console, 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 “Menyunting 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.

Menyunting Kebijakan IAM

Sebuah [kebijakan](#) adalah entitas yang, saat dilampirkan dengan sebuah identitas atau sumber daya, menjelaskan izinnya. Kebijakan disimpan dalam AWS sebagai berkas JSON dan terlampir pada penanggung jawab sebagai kebijakan berbasis identitas dalam IAM. Anda dapat melampirkan kebijakan berbasis identitas ke penanggung jawab (atau identitas), seperti grup pengguna,

pengguna, atau peran IAM. Kebijakan berbasis-identitas dapat mencakup kebijakan terkelola AWS, kebijakan terkelola pelanggan, dan [kebijakan inline](#). Anda dapat menyunting kebijakan terkelola pelanggan dan kebijakan selaras di IAM. AWSkebijakan terkelola tidak dapat disunting. Jumlah dan ukuran sumber daya IAM diAWS akun terbatas. Untuk informasi selengkapnya, lihat [IAM dan AWS STS kuota](#).

Topik

- [Lihat akses kebijakan](#)
- [Menyunting kebijakan terkelola pelanggan \(konsol\)](#)
- [Mengedit kebijakan inline \(konsol\)](#)
- [Mengedit kebijakan terkelola pelanggan \(AWS CLI\)](#)
- [Mengedit kebijakan terkelola pelanggan \(API AWS\)](#)

Lihat akses kebijakan

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 [Menyaring izin di AWS menggunakan informasi yang terakhir diakses](#).


Menyunting kebijakan terkelola pelanggan (konsol)

Anda dapat menyunting kebijakan terkelola pelanggan untuk mengubah izin yang ditetapkan dalam kebijakan. Kebijakan terkelola pelanggan dapat memiliki sampai dengan lima versi. Hal ini penting karena apabila Anda melakukan perubahan pada kebijakan terkelola lebih dari lima versi, AWS Management Console 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](#).

Untuk menyunting kebijakan yang dikelola pelanggan (konsol)

1. Masuk ke AWS Management Console 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 dari berikut:
 - Pilih opsi Visual untuk mengubah kebijakan Anda tanpa pemahaman sintaksis 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 Anda selesai membuat perubahan, pilih Berikutnya untuk melanjutkan.
 - Pilih opsi JSON untuk mengubah kebijakan 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 berpindah antara opsi Editor visual dan JSON kapan pun. Namun, apabila Anda melakukan perubahan atau memilih Berikutnya pada editor visual, IAM dapat merestrukturisasi kebijakan Anda untuk menjadikannya optimal 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. Apabila kebijakan terkelola sudah memiliki maksimum lima versi, memilih Simpan perubahan memunculkan 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.

Untuk mengatur versi default kebijakan terkelola pelanggan (konsol)

1. Masuk ke AWS Management Console 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.

Untuk menghapus versi kebijakan terkelola pelanggan (konsol)

1. Masuk ke AWS Management Console 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 dapat mengedit kebijakan inline dari AWS Management Console.

Untuk mengedit kebijakan inline bagi pengguna, grup pengguna, atau peran (konsol)

1. Di panel navigasi, pilih Pengguna, Grup Pengguna, atau Peran.
2. Dari daftar, pilih nama pengguna, grup pengguna, atau peran dengan kebijakan yang ingin Anda modifikasi. Lalu pilih tab Perizinan dan kembangkan kebijakan.
3. Untuk mengedit kebijakan inline, pilih Edit Policy (Edit Kebijakan).
4. Lakukan salah satu dari berikut:
 - Pilih opsi Visual untuk mengubah kebijakan Anda tanpa pemahaman sintaksis 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 Anda selesai membuat perubahan, pilih Berikutnya untuk melanjutkan.
 - Pilih opsi JSON untuk mengubah kebijakan 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 berpindah antara opsi Editor visual dan JSON kapan pun. Namun, apabila Anda melakukan perubahan atau memilih Berikutnya pada editor visual, IAM dapat merestrukturisasi kebijakan Anda untuk menjadikannya optimal bagi editor visual. Untuk informasi selengkapnya, lihat [Restrukturisasi kebijakan](#).

5. Di halaman Tinjauan, tinjau ringkasan kebijakan lalu pilih Simpan perubahan untuk menyimpan pekerjaan Anda.

Mengedit kebijakan terkelola pelanggan (AWS CLI)

Anda dapat mengedit kebijakan terkelola pelanggan menggunakan AWS Command Line Interface (AWS CLI).

Note

Kebijakan terkelola dapat memiliki sampai dengan 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, grup pengguna, dan peran) yang telah terlampir dengan 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:
 - [memvalidasi-kebijakan](#)

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](#)

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 terkelola pelanggan (API AWS)

Anda dapat mengedit kebijakan terkelola pelanggan menggunakan API AWS.

Note

Kebijakan terkelola dapat memiliki sampai dengan 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 (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, grup pengguna, dan peran) yang telah terlampir dengan 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, panggil operasi IAM Access Analyzer:
 - [ValidatePolicy](#)

Untuk mengatur versi default kebijakan terkelola pelanggan (API AWS)

1. (Opsional) Untuk membuat daftar kebijakan terkelola, hubungi layanan berikut:
 - [ListPolicies](#)

2. Untuk mengatur versi standar kebijakan terkelola pelanggan, hubungi layanan berikut:

- [SetDefaultPolicyVersion](#)

Untuk menghapus suatu versi kebijakan terkelola pelanggan (API AWS)

1. (Opsional) Untuk membuat daftar kebijakan terkelola, hubungi layanan berikut:

- [ListPolicies](#)

2. Untuk menghapus kebijakan terkelola pelanggan, hubungi layanan berikut:

- [DeletePolicyVersion](#)

Menghapus kebijakan IAM

Anda dapat menghapus kebijakan IAM menggunakan AWS Management Console, AWS Command Line Interface (AWS CLI), atau IAM API.

Note

Penghapusan kebijakan IAM bersifat permanen. Setelah kebijakan dihapus kebijakan, kebijakan tidak dapat dipulihkan.

Untuk informasi lebih lanjut tentang perbedaan antara kebijakan terkelola dan kebijakan inline, lihat [Kebijakan terkelola dan kebijakan inline](#).

Untuk informasi umum tentang kebijakan IAM, lihat [Kebijakan dan Izin di IAM](#).

Jumlah dan ukuran sumber daya IAM di AWS akun terbatas. Untuk informasi selengkapnya, lihat [IAM dan AWS STS kuota](#).

Topik

- [Lihat akses kebijakan](#)
- [Menghapus kebijakan IAM \(konsol\)](#)
- [Menghapus kebijakan IAM \(AWS CLI\)](#)
- [Menghapus kebijakan IAM \(AWS API\)](#)

Lihat akses kebijakan

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 [Menyaring izin di AWS menggunakan informasi yang terakhir diakses](#).

Menghapus kebijakan IAM (konsol)

Anda dapat menghapus kebijakan terkelola pelanggan untuk menghilangkannya dari kebijakan terkelola pelanggan Akun AWS. Anda tidak dapat menghapus kebijakan terkelola AWS.

Untuk menyunting kebijakan terkelola pelanggan (konsol)

1. Masuk ke AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Di panel navigasi, pilih Kebijakan.
3. Pilih tombol radio di samping kebijakan terkelola pelanggan untuk menghapus. Anda dapat menggunakan kotak pencarian untuk memfilter daftar kebijakan.
4. Pilih Actions (Tindakan), lalu pilih Delete (Hapus).
5. Ikuti petunjuk untuk mengonfirmasi bahwa Anda ingin menghapus kebijakan, lalu pilih Hapus.

Untuk menghapus kebijakan inline bagi grup pengguna, pengguna, atau peran (konsol)

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 untuk menghapus lalu pilih Hapus. 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 di Pengguna 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**.

Menghapus kebijakan IAM (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, grup pengguna, dan peran) dengan kebijakan terkelola yang terlampir, 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](#)

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 list-user-policies](#)
 - [AWS list-group-policies](#)
 - [AWS 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 get-user-policy](#)

- [AWS get-group-policy](#)
 - [AWS 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 delete-user-policy](#)
 - [AWS delete-group-policy](#)
 - [AWS delete-role-policy](#)

Menghapus kebijakan IAM (AWSAPI)

Anda dapat menghapus kebijakan terkelola pelanggan menggunakan API AWS.

Untuk menghapus kebijakan terkelola pelanggan (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 mendaftarkan identitas (pengguna, grup pengguna, dan peran) dengan kebijakan terkelola terlampir, hubungi layanan 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](#)

Untuk menghapus kebijakan inline (API AWS)

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](#)

Menyaring izin di 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 yang terakhir diakses untuk AWS Organizations](#)
- [Hal yang perlu diketahui tentang informasi yang terakhir diakses](#)
- [Izin diperlukan](#)
- [Pemecahan masalah IAM dan entitas Organisasi](#)
- [Lokasi AWS melacak informasi yang terakhir diakses](#)
- [Melihat informasi yang terakhir diakses untuk IAM](#)
- [Melihat informasi yang terakhir diakses untuk Organisasi](#)
- [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 [Melihat 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 yang terakhir 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 (OU), atau akun. Informasi yang terakhir diakses untuk AWS Organizations mencakup informasi tentang layanan yang diizinkan oleh kebijakan kontrol layanan (Service Control Policy/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 cara melihat informasi

yang terakhir diakses untuk AWS Organizations, lihat [Melihat informasi yang terakhir diakses untuk Organisasi](#).

Sebagai contoh, alur perencanaan penggunaan informasi yang terakhir diakses untuk membuat keputusan tentang izin yang Anda berikan ke entitas organisasi Anda, lihat [Contoh 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 IAM atau entitas Organizations, 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, lihat [Tindakan IAM terakhir mengakses layanan dan tindakan informasi](#). Untuk informasi selengkapnya tentang tindakan Wilayah mana yang terakhir diakses informasi tersedia, lihat [Lokasi AWS melacak informasi yang terakhir diakses](#).
- Upaya dilaporkan — Data layanan yang terakhir diakses mencakup semua upaya untuk mengakses AWS API, bukan hanya upaya yang berhasil. Ini mencakup semua upaya yang dilakukan menggunakan AWS Management Console, API AWS melalui salah satu SDK, 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 data Amazon S3.

- 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 AndaAkun 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 AWS Management Console, Anda mungkin harus menunggunya untuk muncul dan memuat. Apabila Anda menggunakan AWS CLI atau API AWS untuk mendapatkan detail laporan, kredensial Anda harus sesuai dengan kredensial penanggung jawab yang mengeluarkan laporan. Apabila Anda menggunakan kredensial sementara untuk peran atau pengguna 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 pengguna, dan kebijakan) di akun Anda. Informasi terakhir yang diakses untuk Organizations mencakup prinsipal (pengguna IAM, peran IAM, atau) Pengguna root akun AWS dalam entitas Organizations tertentu. 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, AWS Organizations SCP, batas izin IAM, 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 [Menggunakan peran terkait layanan](#) Untuk mempelajari cara jenis kebijakan yang berbeda dievaluasi untuk memungkinkan atau menolak akses, lihat [Logika evaluasi kebijakan](#).
- Jenis kebijakan Organizations — Informasi untuk hanya AWS Organizations mencakup layanan yang diizinkan oleh kebijakan kontrol layanan warisan (SCP) entitas Organizations. SCP merupakan kebijakan yang melekat pada 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 di Organizations, Anda dapat menentukan ID kebijakan secara opsional. Laporan yang dihasilkan mencakup informasi untuk layanan yang diperbolehkan hanya oleh kebijakan tersebut. Informasi ini mencakup aktivitas akun terbaru di entitas Organisasi yang spesifik atau anak dari entitas tersebut. Untuk informasi selengkapnya, lihat [aws iam generate-organizations-access-report](#) atau [GenerateOrganizationsAccessReport](#).
- Akun manajemen organisasi – Anda harus masuk ke akun manajemen organisasi Anda untuk melihat informasi yang terakhir diakses oleh layanan. 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 SCP. 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 entitas Organisasi lain tidak mengembalikan informasi atas aktivitas di akun manajemen.
- Pengaturan Organizations — Administrator harus [mengaktifkan SCP di root organisasi Anda](#) sebelum Anda dapat menghasilkan data untuk Organizations.

Izin diperlukan

Untuk melihat informasi yang terakhir diakses di AWS Management Console, 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.

```
{
  "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 entri root, OU, atau entitas akun dalam Organisasi, 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 untuk 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 melihat layanan informasi yang terakhir diakses untuk Organizations. Selain itu, ini memungkinkan akses baca-saja ke seluruh Organisasi. Kebijakan ini menetapkan izin untuk akses terprogram dan konsol.

```
{
  "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 kebijakan Organizations tertentu. Sebagai contoh kebijakan, lihat [IAM: Lihat informasi yang terakhir diakses untuk kebijakan Organizations yang terakhir diakses untuk](#).

Pemecahan masalah IAM dan entitas Organisasi

Dalam beberapa kejadian, tabel informasi terakhir diakses AWS Management Console Anda mungkin kosong. Atau mungkin permintaan AWS CLI atau API AWS Anda membalikkan set informasi kosong atau kolom 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 [Mengevaluasi kebijakan dalam satu akun](#).
- Untuk kebijakan IAM, pastikan kebijakan pengelolaan yang ditentukan dilampirkan pada setidaknya suatu pengguna, grup dengan anggota, atau peran.
- Untuk entitas Organisasi (root, OU, atau akun), pastikan bahwa Anda masuk menggunakan kredensial akun manajemen Organisasi.
- Verifikasi bahwa [SCP aktif di root organisasi Anda](#).

- Informasi tindakan yang terakhir diakses hanya tersedia untuk tindakan yang tercantum dalam [Tindakan 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 CLI atau API AWS, Anda harus membuat laporan baru untuk melihat informasi yang diperbarui.

Lokasi AWS melacak informasi yang terakhir diakses

AWS mengumpulkan informasi terakhir yang diakses untuk Wilayah AWS standar. Saat AWS menambahkan Wilayah tambahan, Wilayah tersebut ditambahkan ke tabel berikut, termasuk tanggal AWS mulai melacak informasi di masing-masing 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
Afrika (Cape Town)	af-selatan-1	22 April 2020
Asia Pasifik (Hong Kong)	ap-timur-1	24 April 2019
Asia Pacific (Hyderabad)	ap-south-2	November 22, 2022
Asia Pacific (Jakarta)	ap-southeast-3	Desember 13, 2021

Nama wilayah	Region	Tanggal mulai pelacakan
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
Eropa (Paris)	eu-barat-3	18 Desember 2017
Europe (Spain)	eu-south-2	November 15, 2022
Eropa (Stockholm)	eu-utara-1	12 Desember 2018
Europe (Zurich)	eu-central-2	8 November 2022
Israel (Tel Aviv)	il-sentral-1	1 Agustus 2023
Timur Tengah (Bahrain)	me-selatan-1	29 Juli 2019
Timur Tengah (UEA)	eu-central-1	Agustus 30, 2022
Amerika Selatan (Sao Paulo)	sa-timur-1	11 Desember 2015

Nama wilayah	Region	Tanggal mulai pelacakan
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.

Wilayah AWS adalah kumpulan sumber daya AWS di area geografis. Region-region dikelompokkan menjadi beberapa bagian. Region standar adalah region yang termasuk ke bagian aws. Untuk informasi lebih lanjut tentang bagian yang berbeda, lihat [Format Nama Sumber Daya Amazon](#) (Amazon Resource Names/ARN) dalam Referensi Umum AWS. Untuk informasi selengkapnya tentang Wilayah, lihat [Tentang Wilayah AWS](#) dalam Referensi Umum AWS.

Melihat informasi yang terakhir diakses untuk IAM

Anda dapat melihat informasi terakhir yang diakses untuk IAM menggunakan AWS Management Console, 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 [Menyaring izin di 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 yang terakhir diakses untuk IAM di Penasihat Akses di konsol IAM.

Melihat informasi untuk IAM (konsol)

1. Masuk ke AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Di panel navigasi, pilih Grup pengguna, Pengguna, Peran, atau Kebijakan.
3. Pilih pengguna, grup pengguna, peran, atau nama kebijakan untuk halaman Ringkasan dan pilih tab Penasihat Akses. 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 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, lihat [Lokasi AWS melacak informasi yang terakhir 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 lebih lanjut, lihat [Jenis kebijakan](#) atau [Mengevaluasi kebijakan dalam satu akun](#).
 - [aws iam list-policies-granting-service -akses](#)

Melihat informasi untuk IAM (AWSAPI)

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 (AWSAPI)

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 lebih lanjut, lihat [Jenis kebijakan](#) atau [Mengevaluasi kebijakan dalam satu akun](#).
- [ListPoliciesGrantingServiceAccess](#)

Melihat informasi yang terakhir diakses untuk Organisasi

Anda dapat melihat informasi yang terakhir diakses untuk AWS Organizations menggunakan konsol IAM, AWS CLI, atau AWS API. Untuk informasi penting tentang data, izin yang diperlukan, pemecahan masalah, dan region yang didukung lihat [Menyaring izin di AWS menggunakan informasi yang terakhir diakses](#).

Saat Anda masuk ke konsol IAM menggunakan kredensial akun AWS Organizations manajemen, Anda dapat melihat informasi untuk setiap entitas di organisasi Anda. Entitas organisasi mencakup root organisasi, unit organisasi (UO), dan akun-akun. Anda juga dapat menggunakan konsol IAM untuk melihat informasi kebijakan kontrol layanan (SCP) dalam organisasi Anda. IAM menunjukkan daftar layanan yang diizinkan oleh SCPs yang berlaku bagi entitas. Untuk setiap layanan, Anda dapat melihat informasi aktivitas akun terbaru untuk entitas organisasi terpilih atau anak-anak entitas.

Saat Anda menggunakan AWS CLI atau AWS API dengan kredensial akun manajemen, Anda dapat membuat laporan untuk setiap entitas atau kebijakan dalam 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 Organisasi yang spesifik atau cabang dari entitas tersebut.

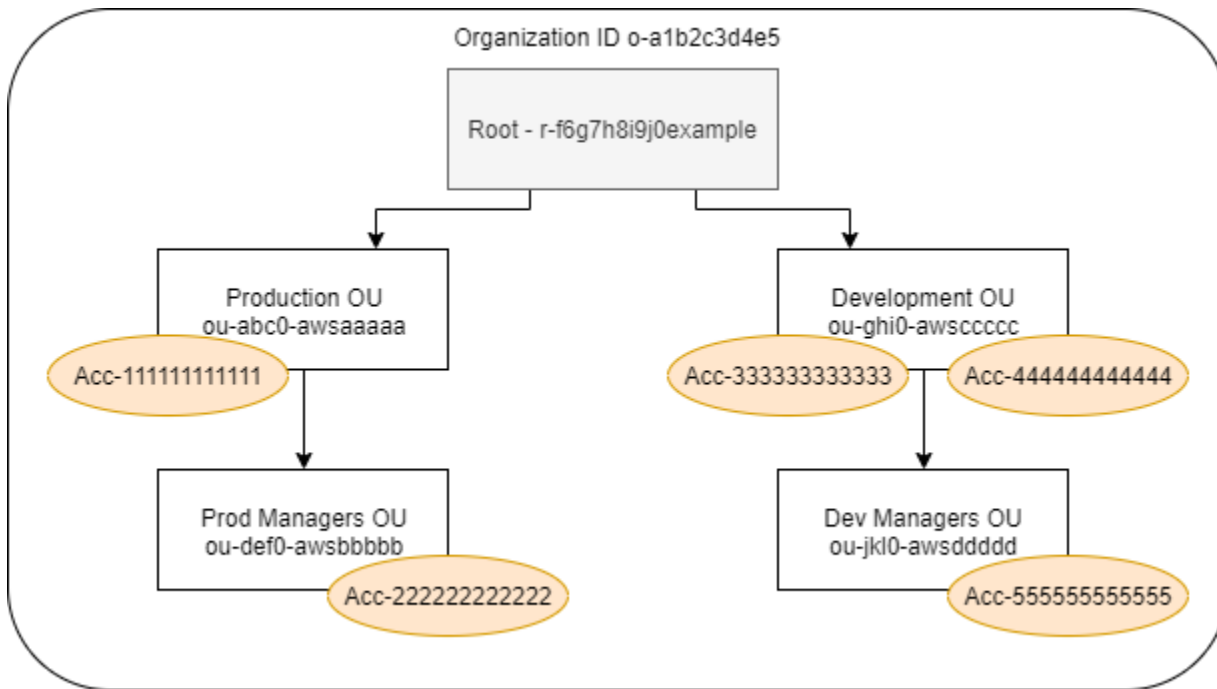
Saat Anda membuat laporan program untuk suatu kebijakan, Anda harus menetapkan entitas Organisasi. Laporan ini mencakup daftar layanan yang diizinkan oleh SCP tertentu. Untuk setiap layanan, ia mencakup aktivitas akun terbaru dalam entitas atau anak entitas yang diberikan izin oleh kebijakan tersebut. Untuk informasi selengkapnya, lihat [aws iam generate-organizations-access-report](#) 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 jalur entitas AWS Organizations

Saat Anda menggunakan AWS CLI atau API AWS untuk membuat laporan akses AWS Organizations, Anda harus menentukan jalur entitas. Jalur adalah representasi teks dari struktur entitas Organisasi.

Anda dapat membangun jalur entitas menggunakan struktur Organisasi Anda yang sudah Anda pahami. Sebagai contoh, anggaplah 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-awsscccc/ou-jkl0-awsdddd/
```

Jalur masuk ke akun di Produksi OU dibangun menggunakan ID dari organisasi, root, OU, dan nomor akun.

```
o-a1b2c3d4e5/r-f6g7h8i9j0example/ou-abc0-awsaaaa/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 Organisasi (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 kredensial akun manajemen Organisasi, dan buka konsol IAM di https://console.aws.amazon.com/iam/](https://console.aws.amazon.com/iam/). [AWS Management Console](#)
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 Terlampir SCPs untuk melihat daftar kebijakan kontrol layanan (SCP) yang dilampirkan pada 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 menyunting SCP di konsol Organisasi. Untuk informasi selengkapnya, lihat [Memperbarui SCP](#) di Panduan AWS Organizations Pengguna.

Untuk melihat informasi untuk OU atau akun (konsol)

1. [Masuk ke kredensial akun manajemen Organisasi, dan buka konsol IAM di https://console.aws.amazon.com/iam/](https://console.aws.amazon.com/iam/). [AWS Management Console](#)
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 Terlampir SCPs untuk melihat daftar kebijakan kontrol layanan (SCP) yang dilampirkan secara 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 menyunting SCP di konsol Organisasi. Untuk informasi selengkapnya, lihat [Memperbarui SCP](#) di Panduan AWS Organizations Pengguna.

Untuk melihat informasi akun manajemen (konsol)

1. [Masuk ke kredensial akun manajemen Organisasi, dan buka konsol IAM di https://console.aws.amazon.com/iam/](https://console.aws.amazon.com/iam/). [AWS Management Console](#)
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 Terlampir SCPs untuk mengonfirmasi bahwa tidak ada lampiran SCP karena akun adalah akun manajemen.

Untuk melihat informasi untuk kebijakan (konsol)

1. [Masuk ke kredensial akun manajemen Organisasi, dan buka konsol IAM di https://console.aws.amazon.com/iam/](https://console.aws.amazon.com/iam/). [AWS Management Console](#)
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 menyunting SCP di konsol Organisasi. Untuk informasi selengkapnya, lihat [Memperbarui SCP](#) di Panduan AWS Organizations Pengguna.

Melihat informasi untuk Organisasi (AWS CLI)

Anda dapat menggunakan AWS CLI untuk mengambil informasi yang terakhir diakses untuk root, OU, akun, atau kebijakan Organisasi Anda.

Untuk melihat informasi layanan Organisasi yang terakhir diakses (AWS CLI)

1. Gunakan kredensial akun manajemen Organisasi Anda dengan izin IAM dan GA yang diperlukan, dan konfirmasi bahwa SCPs diaktifkan untuk root Anda. Untuk informasi selengkapnya, lihat [Hal yang perlu diketahui tentang informasi yang terakhir diakses](#).
2. Buat laporan. Permintaan harus menyertakan alur entitas Organisasi (root, OU, atau akun) yang Anda inginkan laporannya. 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 generate-organizations-access-report](#)
3. Dapatkan detail laporan menggunakan parameter `job-id` dari langkah sebelumnya.
 - [aws iam get-organizations-access-report](#)

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 Organisasi (AWSAPI)

Anda dapat menggunakan AWS API untuk mengambil informasi akses terakhir dari layanan untuk root, OU, akun, atau kebijakan Organisasi Anda.

Untuk melihat informasi layanan Organisasi yang terakhir diakses (AWSAPI)

1. Gunakan kredensial akun manajemen Organisasi Anda dengan izin IAM dan GA yang diperlukan, dan konfirmasi bahwa SCPs diaktifkan untuk root Anda. Untuk informasi selengkapnya, lihat [Hal yang perlu diketahui tentang informasi yang terakhir diakses](#).
2. Buat laporan. Permintaan harus menyertakan alur entitas Organisasi (root, OU, atau akun) yang Anda inginkan laporannya. 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 diakses terakhir untuk membuat keputusan tentang izin yang Anda berikan ke entitas atau AWS Organizations entitas IAM Anda. Untuk informasi selengkapnya, lihat [Menyaring izin di AWS menggunakan informasi yang terakhir diakses](#).

Note

Sebelum Anda melihat informasi akses untuk suatu entitas atau kebijakan di IAM atau AWS Organizations, pastikan bahwa 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 merupakan administrator yang bertanggung jawab untuk menentukan izin pengguna AWS untuk Example Corp. Perusahaan ini baru saja mulai menggunakan AWS, dan tim pengembangan perangkat lunak belum menentukan layanan AWS 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.

```
{  
  
  "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 AWS Management Console. Dia melihat daftar layanan yang diakses anggota grup. Ia mempelajari bahwa pengguna mengakses lima layanan pada minggu lalu: AWS CloudTrail Amazon CloudWatch Logs, Amazon EC2 AWS KMS, dan Amazon S3. Mereka mengakses beberapa layanan lain saat pertama kali mengevaluasi AWS, tetapi tidak sejak saat itu.

Paulo memutuskan untuk mengurangi izin kebijakan dengan hanya menyertakan lima layanan tersebut serta tindakan IAM dan Organisasi 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.

```

{
    "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 [Melihat CloudTrail Peristiwa di CloudTrail Konsol](#) di PanduanAWS 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 IT yang bertanggung jawab untuk memastikan bahwa orang-orang di perusahaannya tidak memiliki izin AWS 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 menandatangani konsol IAM dan memilih Pengguna, memilih nama `nikhilj`, lalu memilih tab Penasihat Akses.

Martha meninjau kolom Terakhir Diakses dan memperhatikan bahwa Nikhil belum mengakses IAM, CloudTrail Route 53, Amazon Elastic Transcoder, dan sejumlah AWS layanan lainnya. Nikhil telah mengakses Amazon S3. Martha memilih S3 dari daftar layanan dan belajar bahwa yang Nikhil telah melakukan beberapa tindakan S3 `List` dalam dua minggu terakhir. Dalam perusahaannya, Martha menegaskan bahwa Nikhil tidak memiliki urusan untuk mengakses IAM dan CloudTrail lagi karena ia bukan lagi 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 yang `IAMFullAccessAWS` dikelola dilampirkan langsung ke `nikhilj` dan dilampirkan ke `security-team` grup.
- CloudTrail - Kebijakan yang `AWSCloudTrailReadOnlyAccessAWS` dikelola melekat pada `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 terkelola `IAMFullAccess AWS` yang dilampirkan langsung ke `nikhilj`. Dia juga menghapus keanggotaan Nikhil ke grup `security-team`. Dua 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 – [Menghapus pengguna](#)
- Grup – [Menghapus grup](#)
- Peran – [Menghapus peran](#)
- Kebijakan – [Menghapus kebijakan terkelola \(ini juga akan menghapus kebijakan dari identitas\)](#)

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 developer dan AWS administrator untuk Example Corp. Ketika timnya mulai menggunakan AWS, mereka memberikan seluruh akses pengguna daya ke semua layanan kecuali IAM dan 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
```

Saat pekerjaan memiliki status COMPLETED, Arnav mengurai isi dari himpunan format JSON 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 mempelajari bahwa `ExamplePolicy1` kebijakan memungkinkan akses ke tiga layanan, Amazon DynamoDB, Amazon S3, dan Amazon EC2. Nama pengguna IAM melakukan upaya `IAMExampleUser` terakhir untuk 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 DynamoDB dan layanan 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.

Sebagai contoh, John Stiles adalah administrator AWS Organizations. Ia bertanggung jawab untuk memastikan bahwa orang-orang dalam izin perusahaan Akun AWS tidak memiliki izin kelebihan. 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. Pada konsol IAM, dia menemukan data Organisasi untuk Development OU. Dia meninjau tabel Laporan akses layanan dan melihat dua layanan AWS yang belum diakses lebih dari periode 180 hari yang dipilihnya. Dia ingat bahwa dia 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. Di AWS Organizations, dia meninjau target untuk mempelajari entitas Organizations yang terlampirkan 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.

Layanan	Awalan layanan
AWS Identity and Access Management Penganalisis Akses	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	application-cost-profiler
Wawasan CloudWatch Aplikasi Amazon	wawasan aplikasi
AWS App Mesh	appmesh
Amazon AppStream 2.0	appstream
AWS AppSync	appsync
Layanan Dikelola Amazon untuk Prometheus	aps

Layanan	Awalan layanan
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
AWS CodeArtifact	kodeartefak
AWS CodeDeploy	penyebaran kode
Amazon CodeGuru Profiler	codeguru-profiler
CodeGuru Peninjau Amazon	pengulas codeguru

Layanan	Awalan layanan
AWS CodePipeline	codepipeline
AWS CodeStar	bintang kode
AWS CodeStar Pemberitahuan	pemberitahuan bintang kode
Identitas Amazon Cognito	kognito-identitas
Kumpulan pengguna Amazon Cognito	kognito-idp
Sinkronisasi Amazon Cognito	sinkronisasi kognito
Amazon Comprehend Medical	memahami medis
AWS Compute Optimizer	pengoptimasi komputasi
AWS Config	config
Amazon Connect	hubungkan
AWS Cost and Usage Report	cur
AWS Glue DataBrew	databrew
AWS Data Exchange	pertukaran data
AWS Data Pipeline	datapipeline
Akselerator DynamoDB	dax
AWS Device Farm	devicefarm
DevOpsGuru Amazon	devops-guru
AWS Direct Connect	DirectConnect
Amazon Data Lifecycle Manager	dlm

Layanan	Awalan layanan
AWS Database Migration Service	dms
Cluster Elastis Amazon DocumentDB	docdb-elastis
AWS Directory Service	ds
Amazon DynamoDB	dynamodb
Toko Blok Elastis Amazon	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 Elastic Inference	inferensi elastis
Amazon ElastiCache	elasticache
AWS Elastic Beanstalk	tangkai beanstalk
Amazon Elastic File System	sistem file elastis
Elastic Load Balancing	elasticlo adbalancing
Amazon Elastic Transcoder	elastictranscoder
Amazon EMR di EKS (Wadah EMR)	kontainer emr
Amazon EMR Tanpa Server	emr-tanpa server
OpenSearch Layanan Amazon	es
Amazon EventBridge	peristiwa

Layanan	Awalan layanan
Amazon CloudWatch Terbukti	ternyata
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
Amazon GameLift	gamelift
Amazon Location Service	geo
Gletser Amazon S3	gletser
Grafana yang Dikelola Amazon	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	lex

Layanan	Awalan layanan
AWS License Manager Manajer Langganan Linux	license-manager-linux-subscriptions
Amazon Lightsail	lightsail
CloudWatch Log Amazon	log
Amazon Lookout for Equipment	peralatan pencarian
Amazon Lookout for Metrics	lookoutmetrics
Amazon Lookout for Vision	lookoutvision
AWS Mainframe Modernization	m2
Blockchain yang Dikelola Amazon	terkelolablockchain
AWS Elemental MediaConnect	mediaconnect
AWS Elemental MediaConvert	mediaconvert
AWS Elemental MediaLive	medialive
AWS Elemental MediaPackage	mediapackage
AWS Elemental MediaPackage VOD	mediapackage-vod
AWS Elemental MediaStore	mediastore
AWS Elemental MediaTailor	mediatailor
Amazon MemoryDB for Redis	memorydb
AWS Application Migration Service	mgn
AWS Migration Hub	mgh
AWS Rekomendasi Strategi Migrasi Hub	strategi hub migrasi

Layanan	Awalan layanan
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 Panorama	panorama
AWS Performance Insights	pi
EventBridgePipa Amazon	pipa
Amazon Polly	polly
Profil 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

Layanan	Awalan layanan
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	pembuat robomak
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 CloudWatchRUM	rum
Layanan Penyimpanan Sederhana Amazon	s3
Amazon S3 on Outposts	pos terdepan s3
Kemampuan SageMaker geospasial Amazon	sagemaker- geospasial
Savings Plans	savingsplans
EventBridgeSkema Amazon	skema

Layanan	Awalan layanan
Amazon SimpleDB	sdb
AWS Secrets Manager	manajer rahasia
AWS Security Hub	securityhub
Danau Keamanan Amazon	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	penandatanganan
AWS SimSpace Weaver	simspaceweaver
AWS Server Migration Service	sms
Amazon Pinpoint SMS dan Layanan Suara	sms-suara
AWS Snowball	bola salju
Layanan Antrian Sederhana Amazon	sqs
AWS Systems Manager	ssm
AWS Systems Manager Incident Manager	insiden ssm
Manajer Sistem AWS untuk SAP	ssm-sap
AWS Step Functions	negara

Layanan	Awalan layanan
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 Telco	tnb
Amazon Transcribe	mentranskripsikan
AWS Transfer Family	transfer
Amazon Translate	menerjemahkan
ID Suara Amazon Connect	voiceid
Kisi VPC Amazon	vpc-kisi
AWS WAFV2	wafv2
AWS Well-Architected Tool	wellarchitected
Kebijaksanaan 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.

Awalan layanan	Tindakan
akses-analyzer	akses-analyzer: ApplyArchiveRule
	akses-analyzer: CancelPolicyGeneration
	akses-analyzer: CheckAccessNotGranted
	akses-analyzer: CheckNoNewAccess
	akses-analyzer: CreateAccessPreview
	akses-analyzer: CreateAnalyzer
	akses-analyzer: CreateArchiveRule
	akses-analyzer: DeleteAnalyzer
	akses-analyzer: DeleteArchiveRule
	akses-analyzer: GetAccessPreview
	akses-analyzer: GetAnalyzedResource
	akses-analyzer: GetAnalyzer
	akses-analyzer: GetArchiveRule
	akses-analyzer: GetFinding
	akses-analyzer: GetGeneratedPolicy
	akses-analyzer: ListAccessPreviewFindings
	akses-analyzer: ListAccessPreviews
	akses-analyzer: ListAnalyzedResources
	akses-analyzer: ListAnalyzers
	akses-analyzer: ListArchiveRules
	akses-analyzer: ListFindings

Awalan layanan	Tindakan
	akses-analyzer: ListPolicyGenerations akses-analyzer: StartPolicyGeneration akses-analyzer: StartResourceScan akses-analyzer: UpdateArchiveRule akses-analyzer: UpdateFindings akses-analyzer: ValidatePolicy
akun	akun: DeleteAlternateContact akun: DisableRegion akun: EnableRegion akun: GetAlternateContact akun: GetContactInformation akun: GetRegionOptStatus akun: ListRegions akun: PutAlternateContact akun: PutContactInformation

Awalan layanan	Tindakan
acm	ACM: DeleteCertificate ACM: DescribeCertificate ACM: ExportCertificate ACM: GetAccountConfiguration ACM: GetCertificate ACM: ImportCertificate ACM: ListCertificates ACM: PutAccountConfiguration ACM: RenewCertificate ACM: RequestCertificate ACM: ResendValidationEmail ACM: UpdateCertificateOptions
aliran udara	aliran udara: CreateCliToken aliran udara: CreateEnvironment aliran udara: CreateWebLoginToken aliran udara: DeleteEnvironment aliran udara: GetEnvironment aliran udara: ListEnvironments 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: GetForm 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: 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: 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: GetApplication appconfig: GetConfiguration appconfig: GetConfigurationProfile appconfig: GetDeployment appconfig: GetDeploymentStrategy appconfig: GetEnvironment appconfig: GetExtension

Awalan layanan	Tindakan
	<p>appconfig: GetExtensionAssociation</p> <p>appconfig: GetHostedConfigurationVersion</p> <p>appconfig: ListApplications</p> <p>appconfig: ListConfigurationProfiles</p> <p>appconfig: ListDeployments</p> <p>appconfig: ListDeploymentStrategies</p> <p>appconfig: ListEnvironments</p> <p>appconfig: ListExtensionAssociations</p> <p>appconfig: ListExtensions</p> <p>appconfig: ListHostedConfigurationVersions</p> <p>appconfig: StartDeployment</p> <p>appconfig: StopDeployment</p> <p>appconfig: UpdateApplication</p> <p>appconfig: UpdateConfigurationProfile</p> <p>appconfig: UpdateDeploymentStrategy</p> <p>appconfig: UpdateEnvironment</p> <p>appconfig: UpdateExtension</p> <p>appconfig: UpdateExtensionAssociation</p> <p>appconfig: ValidateConfiguration</p>

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
application-cost-profiler	application-cost-profiler:DeleteReportDefinition application-cost-profiler:GetReportDefinition application-cost-profiler:ImportApplicationUsage application-cost-profiler:ListReportDefinitions application-cost-profiler:PutReportDefinition application-cost-profiler:UpdateReportDefinition

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: ListLogPatterns

Awalan layanan	Tindakan
	wawasan aplikasi: ListLogPatternSets
	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: BatchAssociateUserStack appstream: BatchDisassociateUserStack appstream: CopyImage appstream: CreateAppBlock appstream: CreateAppBlockBuilder appstream: URL CreateAppBlockBuilderStreaming appstream: CreateApplication appstream: CreateDirectoryConfig appstream: CreateEntitlement appstream: CreateFleet appstream: CreateImageBuilder appstream: URL CreateImageBuilderStreaming appstream: CreateStack appstream: URL CreateStreaming appstream: CreateUpdatedImage appstream: CreateUsageReportSubscription appstream: CreateUser

Awalan layanan	Tindakan
	appstream: DeleteAppBlock
	appstream: DeleteAppBlockBuilder
	appstream: DeleteApplication
	appstream: DeleteDirectoryConfig
	appstream: DeleteEntitlement
	appstream: DeleteFleet
	appstream: DeleteImage
	appstream: DeleteImageBuilder
	appstream: DeleteImagePermissions
	appstream: DeleteStack
	appstream: DeleteUsageReportSubscription
	appstream: DeleteUser
	appstream: DescribeAppBlockBuilderAppBlockAssociations
	appstream: DescribeAppBlockBuilders
	appstream: DescribeAppBlocks
	appstream: DescribeApplicationFleetAssociations
	appstream: DescribeApplications
	appstream: DescribeDirectoryConfigs
	appstream: DescribeEntitlements
	appstream: DescribeFleets
	appstream: DescribeImageBuilders

Awalan layanan	Tindakan
	appstream: DescribeImagePermissions
	appstream: DescribeImages
	appstream: DescribeSessions
	appstream: DescribeStacks
	appstream: DescribeUsageReportSubscriptions
	appstream: DescribeUsers
	appstream: DescribeUserStackAssociations
	appstream: DisableUser
	appstream: DisassociateAppBlockBuilderAppBlock
	appstream: DisassociateApplicationFleet
	appstream: DisassociateApplicationFromEntitlement
	appstream: DisassociateFleet
	appstream: EnableUser
	appstream: ExpireSession
	appstream: ListAssociatedFleets
	appstream: ListAssociatedStacks
	appstream: ListEntitledApplications
	appstream: StartAppBlockBuilder
	appstream: StartFleet
	appstream: StartImageBuilder
	appstream: StopAppBlockBuilder

Awalan layanan	Tindakan
	appstream: StopFleet
	appstream: StopImageBuilder
	appstream: UpdateAppBlockBuilder
	appstream: UpdateApplication
	appstream: UpdateDirectoryConfig
	appstream: UpdateEntitlement
	appstream: UpdateFleet
	appstream: UpdateImagePermissions
	appstream: UpdateStack

Awalan layanan	Tindakan
appsync	appsync: AssociateApi appsync: AssociateMergedGraphQLApi appsync: AssociateSourceGraphQLApi appsync: CreateApiCache appsync: CreateApiKey appsync: CreateDataSource appsync: CreateDomainName appsync: CreateFunction appsync: CreateGraphQLApi appsync: CreateResolver appsync: CreateType appsync: DeleteApiCache appsync: DeleteApiKey appsync: DeleteDataSource appsync: DeleteDomainName appsync: DeleteFunction appsync: DeleteGraphQLApi appsync: DeleteResolver appsync: DeleteType appsync: DisassociateApi appsync: DisassociateMergedGraphQLApi

Awalan layanan	Tindakan
	appsync: DisassociateSourceGraphQLApi
	appsync: EvaluateCode
	appsync: EvaluateMappingTemplate
	appsync: FlushApiCache
	appsync: GetApiAssociation
	appsync: GetApiCache
	appsync: GetDataSource
	appsync: GetDataSourceIntrospection
	appsync: GetDomainName
	appsync: GetFunction
	appsync: GetGraphQLApi
	appsync: GetGraphQLApiEnvironmentVariables
	appsync: GetIntrospectionSchema
	appsync: GetResolver
	appsync: GetSchemaCreationStatus
	appsync: GetSourceApiAssociation
	appsync: GetType
	appsync: ListApiKeys
	appsync: ListDataSources
	appsync: ListDomainNames
	appsync: ListFunctions

Awalan layanan	Tindakan
	appsync: ListGraphQLApis
	appsync: ListResolvers
	appsync: ListResolversByFunction
	appsync: ListSourceApiAssociations
	appsync: ListTypes
	appsync: ListTypesByAssociation
	appsync: PutGraphQLApiEnvironmentVariables
	appsync: StartDataSourceIntrospection
	appsync: StartSchemaCreation
	appsync: StartSchemaMerge
	appsync: UpdateApiCache
	appsync: UpdateApiKey
	appsync: UpdateDataSource
	appsync: UpdateDomainName
	appsync: UpdateFunction
	appsync: UpdateGraphQLApi
	appsync: UpdateResolver
	appsync: UpdateSourceApiAssociation
	appsync: UpdateType

Awalan layanan	Tindakan
aps	aps: CreateAlertManagerDefinition aps: CreateLoggingConfiguration aps: CreateRuleGroupsNamespace aps: CreateScraper aps: CreateWorkspace aps: DeleteAlertManagerDefinition aps: DeleteLoggingConfiguration aps: DeleteRuleGroupsNamespace aps: DeleteScraper aps: DeleteWorkspace aps: DescribeAlertManagerDefinition aps: DescribeLoggingConfiguration aps: DescribeRuleGroupsNamespace aps: DescribeScraper aps: DescribeWorkspace aps: GetDefaultScraperConfiguration aps: ListRuleGroupsNamespaces aps: ListScrapers aps: ListWorkspaces aps: PutAlertManagerDefinition aps: PutRuleGroupsNamespace

Awalan layanan	Tindakan
	aps: UpdateLoggingConfiguration aps: UpdateWorkspaceAlias

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: GetDatabase
	Athena: GetDataCatalog
	Athena: GetNamedQuery
	Athena: GetNotebookMetadata
	Athena: GetPreparedStatement
	Athena: GetQueryExecution
	Athena: GetQueryResults
	Athena: GetQueryResultsStream
	Athena: GetQueryRuntimeStatistics
	Athena: GetSession
	Athena: GetSessionStatus
	Athena: GetTableMetadata
	Athena: GetWorkGroup
	Athena: ImportNotebook
	Athena: ListApplication DPUSizes
	Athena: ListCalculationExecutions
	Athena: ListCapacityReservations
	Athena: ListDatabases
	Athena: ListDataCatalogs

Awalan layanan	Tindakan
	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
	Athena: UpdateNamedQuery
	Athena: UpdateNotebook

Awalan layanan	Tindakan
	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: ListAssessmentFrameworks
	Manajer Audit: ListAssessmentFrameworkShareRequests
	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: AttachLoadBalancers penskalaan otomatis: AttachLoadBalancerTargetGroups 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: DescribeLifecycleHooks
	penskalaan otomatis: DescribeLifecycleHookTypes
	penskalaan otomatis: DescribeLoadBalancers
	penskalaan otomatis: DescribeLoadBalancerTargetGroups
	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: DetachLoadBalancers
	penskalaan otomatis: DetachLoadBalancerTargetGroups
	penskalaan otomatis: DetachTrafficSources
	penskalaan otomatis: DisableMetricsCollection

Awalan layanan	Tindakan
	<p>penskalaan otomatis: EnableMetricsCollection</p> <p>penskalaan otomatis: EnterStandby</p> <p>penskalaan otomatis: ExecutePolicy</p> <p>penskalaan otomatis: ExitStandby</p> <p>penskalaan otomatis: GetPredictiveScalingForecast</p> <p>penskalaan otomatis: PutLifecycleHook</p> <p>penskalaan otomatis: PutNotificationConfiguration</p> <p>penskalaan otomatis: PutScalingPolicy</p> <p>penskalaan otomatis: PutScheduledUpdateGroupAction</p> <p>penskalaan otomatis: PutWarmPool</p> <p>penskalaan otomatis: RecordLifecycleActionHeartbeat</p> <p>penskalaan otomatis: ResumeProcesses</p> <p>penskalaan otomatis: RollbackInstanceRefresh</p> <p>penskalaan otomatis: SetDesiredCapacity</p> <p>penskalaan otomatis: SetInstanceHealth</p> <p>penskalaan otomatis: SetInstanceProtection</p> <p>penskalaan otomatis: StartInstanceRefresh</p> <p>penskalaan otomatis: SuspendProcesses</p> <p>penskalaan otomatis: TerminateInstanceInAutoScalingGroup</p> <p>penskalaan otomatis: UpdateAutoScalingGroup</p>
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: CreateRestoreTestingPlan cadangan: CreateRestoreTestingSelection cadangan: DeleteBackupPlan cadangan: DeleteBackupSelection cadangan: DeleteBackupVault cadangan: DeleteBackupVaultAccessPolicy cadangan: DeleteBackupVaultLockConfiguration cadangan: DeleteBackupVaultNotifications cadangan: DeleteFramework cadangan: DeleteRecoveryPoint cadangan: DeleteReportPlan cadangan: DeleteRestoreTestingPlan cadangan: DeleteRestoreTestingSelection

Awalan layanan	Tindakan
	cadangan: DescribeBackupJob
	cadangan: DescribeBackupVault
	cadangan: DescribeCopyJob
	cadangan: DescribeFramework
	cadangan: DescribeGlobalSettings
	cadangan: DescribeProtectedResource
	cadangan: DescribeRecoveryPoint
	cadangan: DescribeRegionSettings
	cadangan: DescribeReportJob
	cadangan: DescribeReportPlan
	cadangan: DescribeRestoreJob
	cadangan: DisassociateRecoveryPoint
	cadangan: DisassociateRecoveryPointFromParent
	cadangan: ExportBackupPlanTemplate
	cadangan: GetBackupPlan
	cadangan: GetBackupPlanFrom JSON
	cadangan: GetBackupPlanFromTemplate
	cadangan: GetBackupSelection
	cadangan: GetBackupVaultAccessPolicy
	cadangan: GetBackupVaultNotifications
	cadangan: GetLegalHold

Awalan layanan	Tindakan
	cadangan: GetRecoveryPointRestoreMetadata
	cadangan: GetRestoreJobMetadata
	cadangan: GetRestoreTestingInferredMetadata
	cadangan: GetRestoreTestingPlan
	cadangan: GetRestoreTestingSelection
	cadangan: GetSupportedResourceTypes
	cadangan: ListBackupJobs
	cadangan: ListBackupJobSummaries
	cadangan: ListBackupPlans
	cadangan: ListBackupPlanTemplates
	cadangan: ListBackupPlanVersions
	cadangan: ListBackupSelections
	cadangan: ListBackupVaults
	cadangan: ListCopyJobs
	cadangan: ListCopyJobSummaries
	cadangan: ListFrameworks
	cadangan: ListLegalHolds
	cadangan: ListProtectedResources
	cadangan: ListRecoveryPointsByBackupVault
	cadangan: ListRecoveryPointsByLegalHold
	cadangan: ListRecoveryPointsByResource

Awalan layanan	Tindakan
	cadangan: ListReportJobs
	cadangan: ListReportPlans
	cadangan: ListRestoreJobs
	cadangan: ListRestoreJobsByProtectedResource
	cadangan: ListRestoreJobSummaries
	cadangan: ListRestoreTestingPlans
	cadangan: ListRestoreTestingSelections
	cadangan: PutBackupVaultAccessPolicy
	cadangan: PutBackupVaultLockConfiguration
	cadangan: PutBackupVaultNotifications
	cadangan: PutRestoreValidationResult
	cadangan: StartBackupJob
	cadangan: StartCopyJob
	cadangan: StartReportJob
	cadangan: StartRestoreJob
	cadangan: StopBackupJob
	cadangan: UpdateBackupPlan
	cadangan: UpdateFramework
	cadangan: UpdateGlobalSettings
	cadangan: UpdateRecoveryPointLifecycle
	cadangan: UpdateRegionSettings

Awalan layanan	Tindakan
	cadangan: UpdateReportPlan cadangan: UpdateRestoreTestingPlan cadangan: UpdateRestoreTestingSelection

Awalan layanan	Tindakan
batch	batch: CancelJob batch: CreateComputeEnvironment batch: CreateJobQueue batch: CreateSchedulingPolicy batch: DeleteComputeEnvironment batch: DeleteJobQueue batch: DeleteSchedulingPolicy batch: DeregisterJobDefinition batch: DescribeComputeEnvironments batch: DescribeJobDefinitions batch: DescribeJobQueues batch: DescribeJobs batch: DescribeSchedulingPolicies batch: ListJobs batch: ListSchedulingPolicies batch: RegisterJobDefinition batch: SubmitJob batch: TerminateJob batch: UpdateComputeEnvironment batch: UpdateJobQueue batch: UpdateSchedulingPolicy

Awalan layanan	Tindakan
braket	braket: AcceptUserAgreement braket: AccessBraketFeature braket: CancelJob braket: CancelQuantumTask braket: CreateJob braket: CreateQuantumTask braket: GetDevice braket: GetJob braket: GetQuantumTask braket: GetServiceLinkedRoleStatus braket: GetUserAgreementStatus braket: SearchDevices braket: SearchJobs braket: SearchQuantumTasks

Awalan layanan	Tindakan
anggaran	anggaran: ModifyBudget anggaran: CreateBudgetAction anggaran: ModifyBudget anggaran: ModifyBudget anggaran: ModifyBudget anggaran: DeleteBudgetAction anggaran: ModifyBudget anggaran: ModifyBudget anggaran: ViewBudget anggaran: DescribeBudgetAction anggaran: DescribeBudgetActionHistories anggaran: DescribeBudgetActionsForAccount anggaran: DescribeBudgetActionsForBudget anggaran: ViewBudget anggaran: ViewBudget anggaran: ViewBudget anggaran: ViewBudget anggaran: ViewBudget anggaran: ExecuteBudgetAction anggaran: ModifyBudget anggaran: UpdateBudgetAction

Awalan layanan	Tindakan
	anggaran: ModifyBudget anggaran: ModifyBudget
awan9	cloud9: EC2 CreateEnvironment awan9: CreateEnvironmentMembership awan9: DeleteEnvironment awan9: DeleteEnvironmentMembership awan9: DescribeEnvironmentMemberships awan9: DescribeEnvironments awan9: DescribeEnvironmentStatus 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: DescribeGeneratedTemplate pembentukan awan: DescribeOrganizationsAccess pembentukan awan: DescribePublisher

Awalan layanan	Tindakan
	pembentukan awan: DescribeResourceScan
	pembentukan awan: DescribeStackDriftDetectionStatus
	pembentukan awan: DescribeStackEvents
	pembentukan awan: DescribeStackInstance
	pembentukan awan: DescribeStackResource
	pembentukan awan: DescribeStackResourceDrifts
	pembentukan awan: DescribeStackResources
	pembentukan awan: DescribeStacks
	pembentukan awan: DescribeStackSet
	pembentukan awan: DescribeStackSetOperation
	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: GetStackPolicy
	pembentukan awan: GetTemplate
	pembentukan awan: GetTemplateSummary

Awalan layanan	Tindakan
	pembentukan awan: ImportStacksToStackSet
	pembentukan awan: ListChangeSets
	pembentukan awan: ListExports
	pembentukan awan: ListGeneratedTemplates
	pembentukan awan: ListImports
	pembentukan awan: ListResourceScanRelatedResources
	pembentukan awan: ListResourceScanResources
	pembentukan awan: ListResourceScans
	pembentukan awan: ListStackInstanceResourceDrifts
	pembentukan awan: ListStackInstances
	pembentukan awan: ListStackResources
	pembentukan awan: ListStackSetAutoDeploymentTargets
	pembentukan awan: ListStackSetOperationResults
	pembentukan awan: ListStackSetOperations
	pembentukan awan: ListStackSets
	pembentukan awan: ListTypeRegistrations
	pembentukan awan: ListTypes
	pembentukan awan: ListTypeVersions
	pembentukan awan: PublishType
	pembentukan awan: RecordHandlerProgress
	pembentukan awan: RegisterPublisher

Awalan layanan	Tindakan
	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: CreateCachePolicy cloudfront: CreateCloudFrontOriginAccessIdentity cloudfront: CreateContinuousDeploymentPolicy 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: DeleteCachePolicy cloudfront: DeleteCloudFrontOriginAccessIdentity cloudfront: DeleteContinuousDeploymentPolicy cloudfront: DeleteDistribution cloudfront: DeleteFieldLevelEncryptionConfig

Awalan layanan	Tindakan
	cloudfront: DeleteFieldLevelEncryptionProfile
	cloudfront: DeleteFunction
	cloudfront: DeleteKeyGroup
	cloudfront: DeleteKeyValueStore
	cloudfront: DeleteMonitoringSubscription
	cloudfront: DeleteOriginAccessControl
	cloudfront: DeleteOriginRequestPolicy
	cloudfront: DeletePublicKey
	cloudfront: DeleteRealtimeLogConfig
	cloudfront: DeleteResponseHeadersPolicy
	cloudfront: DeleteStreamingDistribution
	cloudfront: DescribeFunction
	cloudfront: DescribeKeyValueStore
	cloudfront: GetCachePolicy
	cloudfront: GetCachePolicyConfig
	cloudfront: GetCloudFrontOriginAccessIdentity
	cloudfront: GetCloudFrontOriginAccessIdentityConfig
	cloudfront: GetContinuousDeploymentPolicy
	cloudfront: GetContinuousDeploymentPolicyConfig
	cloudfront: GetDistributionConfig
	cloudfront: GetFieldLevelEncryption

Awalan layanan	Tindakan
	cloudfront: GetFieldLevelEncryptionConfig
	cloudfront: GetFieldLevelEncryptionProfile
	cloudfront: GetFieldLevelEncryptionProfileConfig
	cloudfront: GetFunction
	cloudfront: GetInvalidation
	cloudfront: GetKeyGroup
	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: ListCachePolicies
	cloudfront: ListCloudFrontOriginAccessIdentities

Awalan layanan	Tindakan
	cloudfront: ListConflictingAliases
	cloudfront: ListContinuousDeploymentPolicies
	cloudfront: ListDistributions
	cloudfront: ListDistributionsByCachePolicyId
	cloudfront: ListDistributionsByKeyGroup
	cloudfront: ListDistributionsByOriginRequestPolicyId
	cloudfront: ListDistributionsByRealtimeLogConfig
	cloudfront: ListDistributionsByResponseHeadersPolicyId
	depan awan: ACLID ListDistributionsByWeb
	cloudfront: ListFieldLevelEncryptionConfigs
	cloudfront: ListFieldLevelEncryptionProfiles
	cloudfront: ListFunctions
	cloudfront: ListInvalidations
	cloudfront: ListKeyGroups
	cloudfront: ListKeyValueStores
	cloudfront: ListOriginAccessControls
	cloudfront: ListOriginRequestPolicies
	cloudfront: ListPublicKeys
	cloudfront: ListRealtimeLogConfigs
	cloudfront: ListResponseHeadersPolicies
	cloudfront: ListStreamingDistributions

Awalan layanan	Tindakan
	<p>cloudfront: PublishFunction</p> <p>cloudfront: TestFunction</p> <p>cloudfront: UpdateCachePolicy</p> <p>cloudfront: UpdateCloudFrontOriginAccessIdentity</p> <p>cloudfront: UpdateContinuousDeploymentPolicy</p> <p>cloudfront: UpdateDistribution</p> <p>cloudfront: UpdateFieldLevelEncryptionConfig</p> <p>cloudfront: UpdateFieldLevelEncryptionProfile</p> <p>cloudfront: UpdateFunction</p> <p>cloudfront: UpdateKeyGroup</p> <p>cloudfront: UpdateKeyValueStore</p> <p>cloudfront: UpdateOriginAccessControl</p> <p>cloudfront: UpdateOriginRequestPolicy</p> <p>cloudfront: UpdatePublicKey</p> <p>cloudfront: UpdateRealtimeLogConfig</p> <p>cloudfront: UpdateResponseHeadersPolicy</p>

Awalan layanan	Tindakan
cloudhsm	cloudhsm: CreateHapg cloudhsm: CreateHsm cloudhsm: CreateLunaClient cloudhsm: DeleteBackup cloudhsm: DeleteHapg cloudhsm: DeleteHsm cloudhsm: DeleteLunaClient cloudhsm: DescribeBackups cloudhsm: DescribeClusters cloudhsm: DescribeHapg cloudhsm: DescribeHsm cloudhsm: DescribeLunaClient cloudhsm: GetConfig cloudhsm: InitializeCluster cloudhsm: ListAvailableZones cloudhsm: ListHapgs cloudhsm: ListHsms cloudhsm: ListLunaClients cloudhsm: ModifyBackupAttributes cloudhsm: ModifyCluster cloudhsm: ModifyHapg

Awalan layanan	Tindakan
	cloudhsm: ModifyHsm
	cloudhsm: ModifyLunaClient
	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
	<p>cloudsearch: ListDomainNames</p> <p>cloudsearch: UpdateAvailabilityOptions</p> <p>cloudsearch: UpdateDomainEndpointOptions</p> <p>cloudsearch: UpdateScalingParameters</p> <p>cloudsearch: UpdateServiceAccessPolicies</p>

Awalan layanan	Tindakan
cloudtrail	cloudtrail: CancelQuery cloudtrail: CreateChannel cloudtrail: CreateEventDataStore cloudtrail: CreateTrail cloudtrail: DeleteChannel cloudtrail: DeleteEventDataStore cloudtrail: DeleteResourcePolicy cloudtrail: DeleteTrail cloudtrail: DeregisterOrganizationDelegatedAdmin cloudtrail: DescribeQuery cloudtrail: DescribeTrails cloudtrail: DisableFederation cloudtrail: GetChannel cloudtrail: GetEventDataStore cloudtrail: GetEventDataStoreData cloudtrail: GetEventSelectors cloudtrail: GetImport cloudtrail: GetInsightSelectors cloudtrail: GetQueryResults cloudtrail: GetResourcePolicy cloudtrail: GetTrail

Awalan layanan	Tindakan
	cloudtrail: GetTrailStatus
	cloudtrail: ListChannels
	cloudtrail: ListEventDataStores
	cloudtrail: ListImportFailures
	cloudtrail: ListImports
	cloudtrail: ListPublicKeys
	cloudtrail: ListQueries
	cloudtrail: ListTrails
	cloudtrail: LookupEvents
	cloudtrail: PutEventSelectors
	cloudtrail: PutInsightSelectors
	cloudtrail: PutResourcePolicy
	cloudtrail: RegisterOrganizationDelegatedAdmin
	cloudtrail: RestoreEventDataStore
	cloudtrail: StartEventDataStoreIngestion
	cloudtrail: StartImport
	cloudtrail: StartLogging
	cloudtrail: StartQuery
	cloudtrail: StopEventDataStoreIngestion
	cloudtrail: StopImport
	cloudtrail: StopLogging

Awalan layanan	Tindakan
	cloudtrail: UpdateChannel
	cloudtrail: UpdateEventDataStore
	cloudtrail: UpdateTrail

Awalan layanan	Tindakan
cloudwatch	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: GetDashboard jam tangan awan: GetInsightRuleReport jam tangan awan: GetMetricStream jam tangan awan: ListDashboards jam tangan awan: ListManagedInsightRules jam tangan awan: ListMetricStreams jam tangan awan: PutAnomalyDetector

Awalan layanan	Tindakan
	jam tangan awan: PutCompositeAlarm
	jam tangan awan: PutDashboard
	jam tangan awan: PutInsightRule
	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: ListPackages
	codeartefak: ListPackageVersionAssets
	codeartefak: ListPackageVersionDependencies
	codeartefak: ListPackageVersions
	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: BatchGetDeployments penyebaran kode: BatchGetDeploymentTargets 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
	<p>penyebaran kode: <code>GetDeploymentConfig</code></p> <p>penyebaran kode: <code>GetDeploymentGroup</code></p> <p>penyebaran kode: <code>GetDeploymentInstance</code></p> <p>penyebaran kode: <code>GetDeploymentTarget</code></p> <p>penyebaran kode: <code>GetOnPremisesInstance</code></p> <p>penyebaran kode: <code>ListApplicationRevisions</code></p> <p>penyebaran kode: <code>ListApplications</code></p> <p>penyebaran kode: <code>ListDeploymentConfigs</code></p> <p>penyebaran kode: <code>ListDeploymentGroups</code></p> <p>penyebaran kode: <code>ListDeploymentInstances</code></p> <p>penyebaran kode: <code>ListDeployments</code></p> <p>penyebaran kode: <code>ListDeploymentTargets</code></p> <p>penyebaran kode: <code>ListGitHubAccountTokenNames</code></p> <p>penyebaran kode: <code>ListOnPremisesInstances</code></p> <p>penyebaran kode: <code>PutLifecycleEventHookExecutionStatus</code></p> <p>penyebaran kode: <code>RegisterApplicationRevision</code></p> <p>penyebaran kode: <code>RegisterOnPremisesInstance</code></p> <p>penyebaran kode: <code>SkipWaitTimeForInstanceTermination</code></p> <p>penyebaran kode: <code>StopDeployment</code></p> <p>penyebaran kode: <code>UpdateApplication</code></p> <p>penyebaran kode: <code>UpdateDeploymentGroup</code></p>

Awalan layanan	Tindakan
codeguru-profiler	codeguru-profiler: AddNotificationChannels
	codeguru-profiler: BatchGetFrameMetricData
	codeguru-profiler: ConfigureAgent
	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: ListWebhooks codepipeline: PollForJobs codepipeline: PollForThirdPartyJobs

Awalan layanan	Tindakan
	codepipeline: PutActionRevision
	codepipeline: PutApprovalResult
	codepipeline: PutJobFailureResult
	codepipeline: PutJobSuccessResult
	codepipeline: PutThirdPartyJobFailureResult
	codepipeline: PutThirdPartyJobSuccessResult
	codepipeline: PutWebhook
	codepipeline: RegisterWebhookWithThirdParty
	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: 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: AdminInitiateAuth cognito-idp: AdminLinkProviderForUser cognito-idp: AdminListDevices cognito-idp: AdminListGroupsWithUser cognito-idp: AdminListUserAuthEvents cognito-idp: AdminRemoveUserFromGroup cognito-idp: AdminResetUserPassword cognito-idp: AdminRespondToAuthChallenge cognito-idp: mfapReference AdminSetUser

Awalan layanan	Tindakan
	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: CreateResourceServer
	cognito-idp: CreateUserImportJob
	cognito-idp: CreateUserPool
	cognito-idp: CreateUserPoolClient
	cognito-idp: CreateUserPoolDomain
	cognito-idp: DeleteGroup
	cognito-idp: DeleteIdentityProvider
	cognito-idp: DeleteResourceServer

Awalan layanan	Tindakan
	cognito-idp: DeleteUser
	cognito-idp: DeleteUserAttributes
	cognito-idp: DeleteUserPool
	cognito-idp: DeleteUserPoolClient
	cognito-idp: DeleteUserPoolDomain
	cognito-idp: DescribeIdentityProvider
	cognito-idp: DescribeResourceServer
	cognito-idp: DescribeRiskConfiguration
	cognito-idp: DescribeUserImportJob
	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: GetSigningCertificate
	Cognito-IDP:getuicustomisasi

Awalan layanan	Tindakan
	cognito-idp: GetUser
	cognito-idp: GetUserAttributeVerificationCode
	cognito-idp: GetUserPoolMfaConfig
	cognito-idp: GlobalSignOut
	cognito-idp: InitiateAuth
	cognito-idp: ListDevices
	cognito-idp: ListGroups
	cognito-idp: ListIdentityProviders
	cognito-idp: ListResourceServers
	cognito-idp: ListUserImportJobs
	cognito-idp: ListUserPoolClients
	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: mfapReference SetUser

Awalan layanan	Tindakan
	cognito-idp: SetUserPoolMfaConfig
	cognito-idp: SetUserSettings
	cognito-idp: SignUp
	cognito-idp: StartUserImportJob
	cognito-idp: StopUserImportJob
	cognito-idp: UpdateAuthEventFeedback
	cognito-idp: UpdateDeviceStatus
	cognito-idp: UpdateGroup
	cognito-idp: UpdateIdentityProvider
	cognito-idp: UpdateResourceServer
	cognito-idp: UpdateUserAttributes
	cognito-idp: UpdateUserPool
	cognito-idp: UpdateUserPoolClient
	cognito-idp: UpdateUserPoolDomain
	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	<p>memahami medis: V2Job DescribeEntitiesDetection</p> <p>ComprehendMedical: Describeicd10cm InferenceJob</p> <p>ComprehendMedical:Describephi DetectionJob</p> <p>memahami medis: DescribeRxNormInferenceJob</p> <p>ComprehendMedical:DescribesNoMEDCT InferenceJob</p> <p>memahami medis: V2 DetectEntities</p> <p>ComprehendMedical: DetectPHI</p> <p>ComprehendMedical: infericd10cm</p> <p>memahami medis: InferRxNorm</p> <p>ComprehendMedical: InfersnomeDCT</p> <p>memahami medis: V2Jobs ListEntitiesDetection</p> <p>ComprehendMedical: listicd10cm InferenceJobs</p> <p>ComprehendMedical:ListPHI DetectionJobs</p> <p>memahami medis: ListRxNormInferenceJobs</p> <p>ComprehendMedical:ListsNomeDCT InferenceJobs</p> <p>memahami medis: V2Job StartEntitiesDetection</p> <p>ComprehendMedical: Starticd10cm InferenceJob</p> <p>ComprehendMedical: StartPHI DetectionJob</p> <p>memahami medis: StartRxNormInferenceJob</p> <p>ComprehendMedical: StartsNomeDCT InferenceJob</p> <p>memahami medis: V2Job StopEntitiesDetection</p>

Awalan layanan	Tindakan
	ComprehendMedical: STOPICD10cm InferenceJob ComprehendMedical: Stopphi DetectionJob memahami medis: StopRxNormInferenceJob ComprehendMedical: StopsnomedCT InferenceJob

Awalan layanan	Tindakan
pengoptimal komputasi	<p data-bbox="544 226 1404 262">pengoptimal komputasi: DeleteRecommendationPreferences</p> <p data-bbox="544 310 1425 346">pengoptimal komputasi: DescribeRecommendationExportJobs</p> <p data-bbox="544 394 1507 430">pengoptimal komputasi: ExportAutoScalingGroupRecommendations</p> <p data-bbox="544 478 1421 514">Pengoptimal komputasi: ExportEbs VolumeRecommendations</p> <p data-bbox="544 562 1445 598">Pengoptimal komputasi: ExportEC2 InstanceRecommendations</p> <p data-bbox="544 646 1412 682">Pengoptimal komputasi: exportecs ServiceRecommendations</p> <p data-bbox="544 730 1485 766">pengoptimal komputasi: ExportLambdaFunctionRecommendations</p> <p data-bbox="544 814 1356 850">pengoptimal komputasi: ExportLicenseRecommendations</p> <p data-bbox="544 898 1502 934">Pengoptimal komputasi: GetEC2 RecommendationProjectedMetrics</p> <p data-bbox="544 982 1485 1039">Pengoptimal komputasi: GETECS ServiceRecommendationProjectedMetrics</p> <p data-bbox="544 1087 1485 1123">pengoptimal komputasi: GetEffectiveRecommendationPreferences</p> <p data-bbox="544 1171 1177 1207">pengoptimal komputasi: GetEnrollmentStatus</p> <p data-bbox="544 1255 1445 1291">pengoptimal komputasi: GetEnrollmentStatusesForOrganization</p> <p data-bbox="544 1339 1364 1375">pengoptimal komputasi: GetRecommendationPreferences</p> <p data-bbox="544 1423 1356 1459">pengoptimal komputasi: GetRecommendationSummaries</p> <p data-bbox="544 1507 1356 1543">pengoptimal komputasi: PutRecommendationPreferences</p> <p data-bbox="544 1591 1234 1627">pengoptimal komputasi: UpdateEnrollmentStatus</p>

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: DescribeConfigurationAggregators
	konfigurasi: DescribeConfigurationAggregatorSourcesStatus
	konfigurasi: DescribeConfigurationRecorders
	konfigurasi: DescribeConfigurationRecorderStatus
	konfigurasi: DescribeConformancePackCompliance
	konfigurasi: DescribeConformancePacks
	konfigurasi: DescribeConformancePackStatus
	konfigurasi: DescribeDeliveryChannels
	konfigurasi: DescribeDeliveryChannelStatus
	konfigurasi: DescribeOrganizationConfigRules
	konfigurasi: DescribeOrganizationConfigRuleStatuses
	konfigurasi: DescribeOrganizationConformancePacks
	konfigurasi: DescribeOrganizationConformancePackStatuses
	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: ListConformancePackComplianceScores
	konfigurasi: ListDiscoveredResources
	konfigurasi: ListResourceEvaluations
	konfigurasi: ListStoredQueries
	konfigurasi: PutConfigRule
	konfigurasi: PutConfigurationAggregator
	konfigurasi: PutConfigurationRecorder

Awalan layanan	Tindakan
	konfigurasi: PutConformancePack
	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: AssociateApprovedOrigin menghubungkan: AssociateBot menghubungkan: AssociateDefaultVocabulary menghubungkan: AssociateFlow menghubungkan: AssociateInstanceStorageConfig menghubungkan: AssociateLambdaFunction menghubungkan: AssociateLexBot menghubungkan: AssociatePhoneNumberContactFlow menghubungkan: AssociateQueueQuickConnects menghubungkan: AssociateRoutingProfileQueues menghubungkan: AssociateSecurityKey menghubungkan: AssociateUserProficiencies menghubungkan: BatchGetFlowAssociation menghubungkan: BatchPutContact menghubungkan: ClaimPhoneNumber menghubungkan: CreateAgentStatus menghubungkan: CreateContactFlow menghubungkan: CreateContactFlowModule menghubungkan: CreateEvaluationForm menghubungkan: CreateHoursOfOperation

Awalan layanan	Tindakan
	<p>menghubungkan: CreateInstance</p> <p>menghubungkan: CreateIntegrationAssociation</p> <p>menghubungkan: CreateParticipant</p> <p>menghubungkan: CreatePersistentContactAssociation</p> <p>menghubungkan: CreatePredefinedAttribute</p> <p>menghubungkan: CreatePrompt</p> <p>menghubungkan: CreateQueue</p> <p>menghubungkan: CreateQuickConnect</p> <p>menghubungkan: CreateRoutingProfile</p> <p>menghubungkan: CreateRule</p> <p>menghubungkan: CreateSecurityProfile</p> <p>menghubungkan: CreateTaskTemplate</p> <p>menghubungkan: CreateTrafficDistributionGroup</p> <p>menghubungkan: CreateUserCase</p> <p>menghubungkan: CreateUser</p> <p>menghubungkan: CreateUserHierarchyGroup</p> <p>menghubungkan: CreateView</p> <p>menghubungkan: CreateViewVersion</p> <p>menghubungkan: CreateVocabulary</p> <p>menghubungkan: DeactivateEvaluationForm</p> <p>menghubungkan: DeleteContactEvaluation</p>

Awalan layanan	Tindakan
	<p>menghubungkan: DeleteContactFlow</p> <p>menghubungkan: DeleteContactFlowModule</p> <p>menghubungkan: DeleteEvaluationForm</p> <p>menghubungkan: DeleteHoursOfOperation</p> <p>menghubungkan: DeleteInstance</p> <p>menghubungkan: DeleteIntegrationAssociation</p> <p>menghubungkan: DeletePredefinedAttribute</p> <p>menghubungkan: DeletePrompt</p> <p>menghubungkan: DeleteQueue</p> <p>menghubungkan: DeleteQuickConnect</p> <p>menghubungkan: DeleteRoutingProfile</p> <p>menghubungkan: DeleteRule</p> <p>menghubungkan: DeleteSecurityProfile</p> <p>menghubungkan: DeleteTaskTemplate</p> <p>menghubungkan: DeleteTrafficDistributionGroup</p> <p>menghubungkan: DeleteUseCase</p> <p>menghubungkan: DeleteUser</p> <p>menghubungkan: DeleteUserHierarchyGroup</p> <p>menghubungkan: DeleteView</p> <p>menghubungkan: DeleteVocabulary</p> <p>menghubungkan: DescribeAgentStatus</p>

Awalan layanan	Tindakan
	<p>menghubungkan: DescribeContactEvaluation</p> <p>menghubungkan: DescribeContactFlow</p> <p>menghubungkan: DescribeContactFlowModule</p> <p>menghubungkan: DescribeEvaluationForm</p> <p>menghubungkan: DescribeInstanceAttribute</p> <p>menghubungkan: DescribeInstanceStorageConfig</p> <p>menghubungkan: DescribePhoneNumber</p> <p>menghubungkan: DescribeRule</p> <p>menghubungkan: DescribeTrafficDistributionGroup</p> <p>menghubungkan: DescribeUserHierarchyGroup</p> <p>menghubungkan: DescribeUserHierarchyStructure</p> <p>menghubungkan: DescribeView</p> <p>menghubungkan: DescribeVocabulary</p> <p>menghubungkan: DisassociateApprovedOrigin</p> <p>menghubungkan: DisassociateBot</p> <p>menghubungkan: DisassociateFlow</p> <p>menghubungkan: DisassociateInstanceStorageConfig</p> <p>menghubungkan: DisassociateLambdaFunction</p> <p>menghubungkan: DisassociateLexBot</p> <p>menghubungkan: DisassociatePhoneNumberContactFlow</p> <p>menghubungkan: DisassociateQueueQuickConnects</p>

Awalan layanan	Tindakan
	<p>menghubungkan: DisassociateRoutingProfileQueues</p> <p>menghubungkan: DisassociateSecurityKey</p> <p>menghubungkan: DisassociateUserProficiencies</p> <p>menghubungkan: DismissUserContact</p> <p>menghubungkan: GetContactAttributes</p> <p>menghubungkan: GetCurrentMetricData</p> <p>menghubungkan: GetCurrentUserData</p> <p>menghubungkan: GetFederationToken</p> <p>menghubungkan: GetFlowAssociation</p> <p>menghubungkan: GetMetricData</p> <p>menghubungkan: GetMetricData V2</p> <p>menghubungkan: GetPromptFile</p> <p>menghubungkan: GetTaskTemplate</p> <p>menghubungkan: GetTrafficDistribution</p> <p>menghubungkan: ImportPhoneNumber</p> <p>menghubungkan: ListApprovedOrigins</p> <p>menghubungkan: ListBots</p> <p>menghubungkan: ListContactEvaluations</p> <p>menghubungkan: ListContactFlowModules</p> <p>menghubungkan: ListContactFlows</p> <p>menghubungkan: ListContactReferences</p>

Awalan layanan	Tindakan
	<p>menghubungkan: ListDefaultVocabularies</p> <p>menghubungkan: ListEvaluationForms</p> <p>menghubungkan: ListEvaluationFormVersions</p> <p>menghubungkan: ListFlowAssociations</p> <p>menghubungkan: ListHoursOfOperations</p> <p>menghubungkan: ListInstanceAttributes</p> <p>menghubungkan: ListInstanceStorageConfigs</p> <p>menghubungkan: ListIntegrationAssociations</p> <p>menghubungkan: ListLambdaFunctions</p> <p>menghubungkan: ListLexBots</p> <p>menghubungkan: ListPhoneNumbers</p> <p>menghubungkan: ListPhoneNumbers V2</p> <p>menghubungkan: ListPredefinedAttributes</p> <p>menghubungkan: ListPrompts</p> <p>menghubungkan: ListQueueQuickConnects</p> <p>menghubungkan: ListQueues</p> <p>menghubungkan: ListQuickConnects</p> <p>menghubungkan: ListRealtimeContactAnalysisSegments V2</p> <p>menghubungkan: ListRoutingProfileQueues</p> <p>menghubungkan: ListRoutingProfiles</p> <p>menghubungkan: ListRules</p>

Awalan layanan	Tindakan
	menghubungkan: ListSecurityKeys
	menghubungkan: ListSecurityProfileApplications
	menghubungkan: ListSecurityProfilePermissions
	menghubungkan: ListSecurityProfiles
	menghubungkan: ListTaskTemplates
	menghubungkan: ListTrafficDistributionGroups
	menghubungkan: ListUseCases
	menghubungkan: ListUserHierarchyGroups
	menghubungkan: ListUserProficiencies
	menghubungkan: ListUsers
	menghubungkan: ListViews
	menghubungkan: ListViewVersions
	menghubungkan: MonitorContact
	menghubungkan: PauseContact
	menghubungkan: PutUserStatus
	menghubungkan: ReleasePhoneNumber
	menghubungkan: ReplicateInstance
	menghubungkan: ResumeContact
	menghubungkan: ResumeContactRecording
	menghubungkan: SearchAvailablePhoneNumbers
	menghubungkan: SearchContacts

Awalan layanan	Tindakan
	<p>menghubungkan: SearchHoursOfOperations</p> <p>menghubungkan: SearchPredefinedAttributes</p> <p>menghubungkan: SearchPrompts</p> <p>menghubungkan: SearchQueues</p> <p>menghubungkan: SearchQuickConnects</p> <p>menghubungkan: SearchRoutingProfiles</p> <p>menghubungkan: SearchSecurityProfiles</p> <p>menghubungkan: SearchVocabularies</p> <p>menghubungkan: SendChatIntegrationEvent</p> <p>menghubungkan: StartChatContact</p> <p>menghubungkan: StartContactEvaluation</p> <p>menghubungkan: StartContactRecording</p> <p>menghubungkan: StartContactStreaming</p> <p>menghubungkan: StartOutboundVoiceContact</p> <p>menghubungkan: StartTaskContact</p> <p>menghubungkan: StartWeb RTCCContact</p> <p>menghubungkan: StopContact</p> <p>menghubungkan: StopContactRecording</p> <p>menghubungkan: StopContactStreaming</p> <p>menghubungkan: SubmitContactEvaluation</p> <p>menghubungkan: SuspendContactRecording</p>

Awalan layanan	Tindakan
	<p>menghubungkan: TransferContact</p> <p>menghubungkan: UpdateAgentStatus</p> <p>menghubungkan: UpdateContact</p> <p>menghubungkan: UpdateContactAttributes</p> <p>menghubungkan: UpdateContactEvaluation</p> <p>menghubungkan: UpdateContactFlowContent</p> <p>menghubungkan: UpdateContactFlowMetadata</p> <p>menghubungkan: UpdateContactFlowModuleContent</p> <p>menghubungkan: UpdateContactFlowModuleMetadata</p> <p>menghubungkan: UpdateContactFlowName</p> <p>menghubungkan: UpdateContactRoutingData</p> <p>menghubungkan: UpdateContactSchedule</p> <p>menghubungkan: UpdateEvaluationForm</p> <p>menghubungkan: UpdateHoursOfOperation</p> <p>menghubungkan: UpdateInstanceAttribute</p> <p>menghubungkan: UpdateInstanceStorageConfig</p> <p>menghubungkan: UpdateParticipantRoleConfig</p> <p>menghubungkan: UpdatePhoneNumber</p> <p>menghubungkan: UpdatePhoneNumberMetadata</p> <p>menghubungkan: UpdatePredefinedAttribute</p> <p>menghubungkan: UpdatePrompt</p>

Awalan layanan	Tindakan
	<p>menghubungkan: UpdateQueueHoursOfOperation</p> <p>menghubungkan: UpdateQueueMaxContacts</p> <p>menghubungkan: UpdateQueueName</p> <p>menghubungkan: UpdateQueueOutboundCallerConfig</p> <p>menghubungkan: UpdateQueueStatus</p> <p>menghubungkan: UpdateQuickConnectConfig</p> <p>menghubungkan: UpdateQuickConnectName</p> <p>menghubungkan: UpdateRoutingProfileAgentAvailabilityTimer</p> <p>menghubungkan: UpdateRoutingProfileConcurrency</p> <p>menghubungkan: UpdateRoutingProfileDefaultOutboundQueue</p> <p>menghubungkan: UpdateRoutingProfileName</p> <p>menghubungkan: UpdateRoutingProfileQueues</p> <p>menghubungkan: UpdateRule</p> <p>menghubungkan: UpdateSecurityProfile</p> <p>menghubungkan: UpdateTaskTemplate</p> <p>menghubungkan: UpdateTrafficDistribution</p> <p>menghubungkan: UpdateUserHierarchy</p> <p>menghubungkan: UpdateUserHierarchyGroupName</p> <p>menghubungkan: UpdateUserHierarchyStructure</p> <p>menghubungkan: UpdateUserIdentityInfo</p> <p>menghubungkan: UpdateUserPhoneConfig</p>

Awalan layanan	Tindakan
	menghubungkan: UpdateUserProficiencies menghubungkan: UpdateUserRoutingProfile menghubungkan: UpdateUserSecurityProfiles menghubungkan: UpdateViewContent menghubungkan: UpdateViewMetadata
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: ListRecipes
	Databrew: ListRecipeVersions
	Databrew: ListRulesets
	Databrew: ListSchedules
	Databrew: PublishRecipe
	Databrew: SendProjectSessionAction
	Databrew: StartJobRun
	Databrew: StartProjectSession
	Databrew: StopJobRun
	Databrew: UpdateDataset
	Databrew: UpdateProfileJob
	Databrew: UpdateProject
	Databrew: UpdateRecipe
	Databrew: UpdateRecipeJob
	Databrew: UpdateRuleset
	Databrew: UpdateSchedule

Awalan layanan	Tindakan
pertukaran data	pertukaran data: CancelJob pertukaran data: CreateDataSet pertukaran data: CreateEventAction pertukaran data: CreateJob pertukaran data: CreateRevision pertukaran data: DeleteAsset pertukaran data: DeleteEventAction pertukaran data: DeleteRevision pertukaran data: GetEventAction pertukaran data: GetJob pertukaran data: ListDataSetRevisions pertukaran data: ListDataSets pertukaran data: ListEventActions pertukaran data: ListJobs pertukaran data: ListRevisionAssets pertukaran data: RevokeRevision pertukaran data: SendDataSetNotification pertukaran data: StartJob pertukaran data: UpdateAsset pertukaran data: UpdateDataSet pertukaran data: UpdateEventAction

Awalan layanan	Tindakan
	pertukaran data: UpdateRevision
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
	<p>devicefarm: ListNetworkProfiles</p> <p>devicefarm: ListOfferingPromotions</p> <p>devicefarm: ListOfferings</p> <p>devicefarm: ListOfferingTransactions</p> <p>devicefarm: ListProjects</p> <p>devicefarm: ListRemoteAccessSessions</p> <p>devicefarm: ListRuns</p> <p>devicefarm: ListSamples</p> <p>devicefarm: ListSuites</p> <p>devicefarm: ListTestGridProjects</p> <p>devicefarm: ListTestGridSessionActions</p> <p>devicefarm: ListTestGridSessionArtifacts</p> <p>devicefarm: ListTestGridSessions</p> <p>devicefarm: ListTests</p> <p>devicefarm: ListUniqueProblems</p> <p>devicefarm: ListUploads</p> <p>DeviceFarm:ListVPCEConfigurations</p> <p>devicefarm: PurchaseOffering</p> <p>devicefarm: RenewOffering</p> <p>devicefarm: ScheduleRun</p> <p>devicefarm: StopJob</p>

Awalan layanan	Tindakan
	<p>devicefarm: StopRemoteAccessSession</p> <p>devicefarm: StopRun</p> <p>devicefarm: UpdateDeviceInstance</p> <p>devicefarm: UpdateDevicePool</p> <p>devicefarm: UpdateInstanceProfile</p> <p>devicefarm: UpdateNetworkProfile</p> <p>devicefarm: UpdateProject</p> <p>devicefarm: UpdateTestGridProject</p> <p>devicefarm: UpdateUpload</p> <p>DeviceFarm:updatevpceconfiguration</p>

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	<p>directconnect: AcceptDirectConnectGatewayAssociationProposal</p> <p>directconnect: AllocateConnectionOnInterconnect</p> <p>directconnect: AllocateHostedConnection</p> <p>directconnect: AllocatePrivateVirtualInterface</p> <p>directconnect: AllocatePublicVirtualInterface</p> <p>directconnect: AllocateTransitVirtualInterface</p> <p>directconnect: AssociateConnectionWithLag</p> <p>directconnect: AssociateHostedConnection</p> <p>directconnect: AssociateMacSecKey</p> <p>directconnect: AssociateVirtualInterface</p> <p>directconnect: ConfirmConnection</p> <p>directconnect: ConfirmCustomerAgreement</p> <p>directconnect: ConfirmPrivateVirtualInterface</p> <p>directconnect: ConfirmPublicVirtualInterface</p> <p>directconnect: ConfirmTransitVirtualInterface</p> <p>DirectConnect:CreateBGPpeer</p> <p>directconnect: CreateConnection</p> <p>directconnect: CreateDirectConnectGateway</p> <p>directconnect: CreateDirectConnectGatewayAssociation</p> <p>directconnect: CreateDirectConnectGatewayAssociationProposal</p> <p>directconnect: CreateInterconnect</p>

Awalan layanan	Tindakan
	<p>directconnect: CreateLag</p> <p>directconnect: CreatePrivateVirtualInterface</p> <p>directconnect: CreatePublicVirtualInterface</p> <p>directconnect: CreateTransitVirtualInterface</p> <p>DirectConnect:DeleteBGPPeer</p> <p>directconnect: DeleteConnection</p> <p>directconnect: DeleteDirectConnectGateway</p> <p>directconnect: DeleteDirectConnectGatewayAssociation</p> <p>directconnect: DeleteDirectConnectGatewayAssociationProposal</p> <p>directconnect: DeleteInterconnect</p> <p>directconnect: DeleteLag</p> <p>directconnect: DeleteVirtualInterface</p> <p>directconnect: DescribeConnectionLoa</p> <p>directconnect: DescribeConnections</p> <p>directconnect: DescribeConnectionsOnInterconnect</p> <p>directconnect: DescribeCustomerMetadata</p> <p>directconnect: DescribeDirectConnectGatewayAssociationProposals</p> <p>directconnect: DescribeDirectConnectGatewayAssociations</p> <p>directconnect: DescribeDirectConnectGatewayAttachments</p> <p>directconnect: DescribeDirectConnectGateways</p>

Awalan layanan	Tindakan
	<p>directconnect: DescribeHostedConnections</p> <p>directconnect: DescribeInterconnectLoa</p> <p>directconnect: DescribeInterconnects</p> <p>directconnect: DescribeLags</p> <p>directconnect: DescribeLoa</p> <p>directconnect: DescribeLocations</p> <p>directconnect: DescribeRouterConfiguration</p> <p>directconnect: DescribeVirtualGateways</p> <p>directconnect: DescribeVirtualInterfaces</p> <p>directconnect: DisassociateConnectionFromLag</p> <p>directconnect: DisassociateMacSecKey</p> <p>directconnect: ListVirtualInterfaceTestHistory</p> <p>directconnect: StartBgpFailoverTest</p> <p>directconnect: StopBgpFailoverTest</p> <p>directconnect: UpdateConnection</p> <p>directconnect: UpdateDirectConnectGateway</p> <p>directconnect: UpdateDirectConnectGatewayAssociation</p> <p>directconnect: UpdateLag</p> <p>directconnect: UpdateVirtualInterfaceAttributes</p>

Awalan layanan	Tindakan
dIm	dIm: CreateLifecyclePolicy dIm: DeleteLifecyclePolicy dIm: GetLifecyclePolicies dIm: GetLifecyclePolicy dIm: UpdateLifecyclePolicy

Awalan layanan	Tindakan
dms	dms: ApplyPendingMaintenanceAction dms: BatchStartRecommendations 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: DeleteDataProvider dms: DeleteEndpoint dms: DeleteEventSubscription dms: DeleteFleetAdvisorCollector dms: DeleteFleetAdvisorDatabases dms: DeleteInstanceProfile dms: DeleteMigrationProject

Awalan layanan	Tindakan
	<p>dms: DeleteReplicationConfig</p> <p>dms: DeleteReplicationInstance</p> <p>dms: DeleteReplicationSubnetGroup</p> <p>dms: DeleteReplicationTask</p> <p>dms: DeleteReplicationTaskAssessmentRun</p> <p>dms: DescribeAccountAttributes</p> <p>dms: DescribeApplicableIndividualAssessments</p> <p>dms: DescribeCertificates</p> <p>dms: DescribeConnections</p> <p>dms: DescribeEndpoints</p> <p>dms: DescribeEndpointSettings</p> <p>dms: DescribeEndpointTypes</p> <p>dms: DescribeEngineVersions</p> <p>dms: DescribeEventCategories</p> <p>dms: DescribeEvents</p> <p>dms: DescribeEventSubscriptions</p> <p>dms: DescribeFleetAdvisorCollectors</p> <p>dms: DescribeFleetAdvisorDatabases</p> <p>dms: DescribeFleetAdvisorLsaAnalysis</p> <p>dms: DescribeFleetAdvisorSchemaObjectSummary</p> <p>dms: DescribeFleetAdvisorSchemas</p>

Awalan layanan	Tindakan
	<p>dms: DescribeMetadataModelImports</p> <p>dms: DescribeOrderableReplicationInstances</p> <p>dms: DescribePendingMaintenanceActions</p> <p>dms: DescribeRecommendationLimitations</p> <p>dms: DescribeRecommendations</p> <p>dms: DescribeRefreshSchemasStatus</p> <p>dms: DescribeReplicationConfigs</p> <p>dms: DescribeReplicationInstances</p> <p>dms: DescribeReplicationInstanceTaskLogs</p> <p>dms: DescribeReplications</p> <p>dms: DescribeReplicationSubnetGroups</p> <p>dms: DescribeReplicationTableStatistics</p> <p>dms: DescribeReplicationTaskAssessmentResults</p> <p>dms: DescribeReplicationTaskAssessmentRuns</p> <p>dms: DescribeReplicationTaskIndividualAssessments</p> <p>dms: DescribeReplicationTasks</p> <p>dms: DescribeSchemas</p> <p>dms: DescribeTableStatistics</p> <p>dms: ExportMetadataModelAssessment</p> <p>dms: GetMetadataModel</p> <p>dms: ImportCertificate</p>

Awalan layanan	Tindakan
	<p>dms: ListMetadataModelAssessmentActionItems</p> <p>dms: ModifyEndpoint</p> <p>dms: ModifyEventSubscription</p> <p>dms: ModifyReplicationConfig</p> <p>dms: ModifyReplicationInstance</p> <p>dms: ModifyReplicationSubnetGroup</p> <p>dms: ModifyReplicationTask</p> <p>dms: MoveReplicationTask</p> <p>dms: RebootReplicationInstance</p> <p>dms: RefreshSchemas</p> <p>dms: ReloadReplicationTables</p> <p>dms: ReloadTables</p> <p>dms: RunFleetAdvisorLsaAnalysis</p> <p>dms: StartMetadataModelAssessment</p> <p>dms: StartMetadataModelConversion</p> <p>dms: StartMetadataModelExportToTarget</p> <p>dms: StartRecommendations</p> <p>dms: StartReplication</p> <p>dms: StartReplicationTask</p> <p>dms: StartReplicationTaskAssessment</p> <p>dms: StopReplicationTask</p>

Awalan layanan	Tindakan
	dms: TestConnection dms: UpdateSubscriptionsToEventBridge
docdb-elastic	docdb-elastic: CopyClusterSnapshot docdb-elastic: DeleteCluster docdb-elastic: DeleteClusterSnapshot docdb-elastic: GetCluster docdb-elastic: GetClusterSnapshot docdb-elastic: ListClusters docdb-elastic: ListClusterSnapshots docdb-elastic: RestoreClusterFromSnapshot docdb-elastic: StartCluster docdb-elastic: StopCluster docdb-elastic: UpdateCluster

Awalan layanan	Tindakan
ds	ds: AcceptSharedDirectory
	ds: AddIpRoutes
	ds: AddRegion
	ds: CancelSchemaExtension
	ds: ConnectDirectory
	ds: CreateAlias
	ds: CreateComputer
	ds: CreateConditionalForwarder
	ds: CreateDirectory
	ds: CreateLogSubscription
	ds: CreateMicrosoft AD
	ds: CreateSnapshot
	ds: CreateTrust
	ds: DeleteConditionalForwarder
	ds: DeleteDirectory
	ds: DeleteLogSubscription
	ds: DeleteSnapshot
	ds: DeleteTrust
	ds: DeregisterCertificate
	ds: DeregisterEventTopic
	ds: DescribeCertificate

Awalan layanan	Tindakan
	<p>ds: DescribeClientAuthenticationSettings</p> <p>ds: DescribeConditionalForwarders</p> <p>ds: DescribeDirectories</p> <p>ds: DescribeDomainControllers</p> <p>ds: DescribeEventTopics</p> <p>DS: DescribeldapsSettings</p> <p>ds: DescribeRegions</p> <p>ds: DescribeSettings</p> <p>ds: DescribeSharedDirectories</p> <p>ds: DescribeSnapshots</p> <p>ds: DescribeTrusts</p> <p>ds: DescribeUpdateDirectory</p> <p>ds: DisableClientAuthentication</p> <p>DS: Nonaktifkan DAPS</p> <p>ds: DisableRadius</p> <p>ds: DisableSso</p> <p>ds: EnableClientAuthentication</p> <p>DS: EnableIDAPS</p> <p>ds: EnableRadius</p> <p>ds: EnableSso</p> <p>ds: GetDirectoryLimits</p>

Awalan layanan	Tindakan
	ds: GetSnapshotLimits
	ds: ListCertificates
	ds: ListIpRoutes
	ds: ListLogSubscriptions
	ds: ListSchemaExtensions
	ds: RegisterCertificate
	ds: RegisterEventTopic
	ds: RejectSharedDirectory
	ds: RemoveIpRoutes
	ds: RemoveRegion
	ds: ResetUserPassword
	ds: RestoreFromSnapshot
	ds: ShareDirectory
	ds: StartSchemaExtension
	ds: UnshareDirectory
	ds: UpdateConditionalForwarder
	ds: UpdateDirectorySetup
	ds: UpdateNumberOfDomainControllers
	ds: UpdateRadius
	ds: UpdateSettings
	ds: UpdateTrust

Awalan layanan	Tindakan
	ds: VerifyTrust

Awalan layanan	Tindakan
dynamodb	dynamodb: CreateBackup dynamodb: CreateGlobalTable dynamodb: CreateTable dynamodb: DeleteBackup dynamodb: DeleteTable dynamodb: DescribeBackup dynamodb: DescribeContinuousBackups dynamodb: DescribeContributorInsights dynamodb: DescribeEndpoints dynamodb: DescribeExport dynamodb: DescribeGlobalTable dynamodb: DescribeGlobalTableSettings dynamodb: DescribeImport dynamodb: DescribeKinesisStreamingDestination dynamodb: DescribeLimits dynamodb: DescribeStream dynamodb: DescribeTable dynamodb: DescribeTableReplicaAutoScaling dynamodb: DescribeTimeToLive dynamodb: DisableKinesisStreamingDestination dynamodb: EnableKinesisStreamingDestination

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: AcceptReservedInstancesExchangeQuote EC2: AcceptTransitGatewayMulticastDomainAssociations EC2: AcceptTransitGatewayPeeringAttachment EC2: AcceptTransitGatewayVpcAttachment EC2: AcceptVpcEndpointConnections EC2: AcceptVpcPeeringConnection EC2: AdvertiseByoipCidr EC2: AllocateAddress EC2: AllocateHosts EC2: AllocateIpamPoolCidr EC2: ApplySecurityGroupsToClientVpnTargetNetwork ec2:6Alamat AssignIpv EC2: AssignPrivateIpAddresses EC2: AssignPrivateNatGatewayAddress EC2: AssociateAddress EC2: AssociateClientVpnTargetNetwork EC2: AssociateDhcpOptions EC2: AssociateEnclaveCertificateIamRole EC2: AssociateIamInstanceProfile EC2: AssociateInstanceEventWindow

Awalan layanan	Tindakan
	EC2: AssociateIpamByoasn
	EC2: AssociateIpamResourceDiscovery
	EC2: AssociateNatGatewayAddress
	EC2: AssociateRouteTable
	EC2: AssociateSubnetCidrBlock
	EC2: AssociateTransitGatewayMulticastDomain
	EC2: AssociateTransitGatewayPolicyTable
	EC2: AssociateTransitGatewayRouteTable
	EC2: AssociateTrunkInterface
	EC2: AssociateVpcCidrBlock
	EC2: AttachClassicLinkVpc
	EC2: AttachInternetGateway
	EC2: AttachNetworkInterface
	EC2: AttachVerifiedAccessTrustProvider
	EC2: AttachVolume
	EC2: AttachVpnGateway
	EC2: AuthorizeClientVpnIngress
	EC2: AuthorizeSecurityGroupEgress
	EC2: AuthorizeSecurityGroupIngress
	EC2: BundleInstance
	EC2: CancelBundleTask

Awalan layanan	Tindakan
	EC2: CancelCapacityReservation
	EC2: CancelCapacityReservationFleets
	EC2: CancelConversionTask
	EC2: CancelExportTask
	EC2: CancellImageLaunchPermission
	EC2: CancellImportTask
	EC2: CancelReservedInstancesListing
	EC2: CancelSpotFleetRequests
	EC2: CancelSpotInstanceRequests
	EC2: ConfirmProductInstance
	EC2: CopyFpgaImage
	EC2: CopyImage
	EC2: CopySnapshot
	EC2: CreateCapacityReservation
	EC2: CreateCapacityReservationFleet
	EC2: CreateCarrierGateway
	EC2: CreateClientVpnEndpoint
	EC2: CreateClientVpnRoute
	EC2: CreateCoipCidr
	EC2: CreateCoipPool
	EC2: CreateCustomerGateway

Awalan layanan	Tindakan
	EC2: CreateDefaultSubnet
	EC2: CreateDefaultVpc
	EC2: CreateDhcpOptions
	EC2: CreateEgressOnlyInternetGateway
	EC2: CreateFleet
	EC2: CreateFlowLogs
	EC2: CreateFpgaImage
	EC2: CreateImage
	EC2: CreateInstanceConnectEndpoint
	EC2: CreateInstanceEventWindow
	EC2: CreateInstanceExportTask
	EC2: CreateInternetGateway
	EC2: CreateIam
	EC2: CreateIamPool
	EC2: CreateIamResourceDiscovery
	EC2: CreateIamScope
	EC2: CreateKeyPair
	EC2: CreateLaunchTemplate
	EC2: CreateLaunchTemplateVersion
	EC2: CreateLocalGatewayRoute
	EC2: CreateLocalGatewayRouteTable

Awalan layanan	Tindakan
	EC2: CreateLocalGatewayRouteTableVirtualInterfaceGroupAssociation
	EC2: CreateLocalGatewayRouteTableVpcAssociation
	EC2: CreateManagedPrefixList
	EC2: CreateNatGateway
	EC2: CreateNetworkAcl
	EC2: CreateNetworkAclEntry
	EC2: CreateNetworkInsightsAccessScope
	EC2: CreateNetworkInsightsPath
	EC2: CreateNetworkInterface
	EC2: CreateNetworkInterfacePermission
	EC2: CreatePlacementGroup
	ec2:4Kolam CreatePublicIpv
	EC2: CreateReplaceRootVolumeTask
	EC2: CreateReservedInstancesListing
	EC2: CreateRestoreImageTask
	EC2: CreateRoute
	EC2: CreateRouteTable
	EC2: CreateSecurityGroup
	EC2: CreateSnapshot
	EC2: CreateSnapshots

Awalan layanan	Tindakan
	EC2: CreateSpotDatafeedSubscription
	EC2: CreateStoreImageTask
	EC2: CreateSubnet
	EC2: CreateSubnetCidrReservation
	EC2: CreateTrafficMirrorFilter
	EC2: CreateTrafficMirrorFilterRule
	EC2: CreateTrafficMirrorSession
	EC2: CreateTrafficMirrorTarget
	EC2: CreateTransitGateway
	EC2: CreateTransitGatewayConnect
	EC2: CreateTransitGatewayConnectPeer
	EC2: CreateTransitGatewayMulticastDomain
	EC2: CreateTransitGatewayPeeringAttachment
	EC2: CreateTransitGatewayPolicyTable
	EC2: CreateTransitGatewayPrefixListReference
	EC2: CreateTransitGatewayRoute
	EC2: CreateTransitGatewayRouteTable
	EC2: CreateTransitGatewayRouteTableAnnouncement
	EC2: CreateTransitGatewayVpcAttachment
	EC2: CreateVerifiedAccessEndpoint
	EC2: CreateVerifiedAccessGroup

Awalan layanan	Tindakan
	EC2: CreateVerifiedAccessInstance
	EC2: CreateVerifiedAccessTrustProvider
	EC2: CreateVolume
	EC2: CreateVpc
	EC2: CreateVpcEndpoint
	EC2: CreateVpcEndpointConnectionNotification
	EC2: CreateVpcEndpointServiceConfiguration
	EC2: CreateVpcPeeringConnection
	EC2: CreateVpnConnection
	EC2: CreateVpnConnectionRoute
	EC2: CreateVpnGateway
	EC2: DeleteCarrierGateway
	EC2: DeleteClientVpnEndpoint
	EC2: DeleteClientVpnRoute
	EC2: DeleteCoipCidr
	EC2: DeleteCoipPool
	EC2: DeleteCustomerGateway
	EC2: DeleteDhcpOptions
	EC2: DeleteEgressOnlyInternetGateway
	EC2: DeleteFleets
	EC2: DeleteFlowLogs

Awalan layanan	Tindakan
	EC2: DeleteFpgaImage
	EC2: DeleteInstanceConnectEndpoint
	EC2: DeleteInstanceEventWindow
	EC2: DeleteInternetGateway
	EC2: DeleteIpam
	EC2: DeleteIpamPool
	EC2: DeleteIpamResourceDiscovery
	EC2: DeleteIpamScope
	EC2: DeleteKeyPair
	EC2: DeleteLaunchTemplate
	EC2: DeleteLaunchTemplateVersions
	EC2: DeleteLocalGatewayRoute
	EC2: DeleteLocalGatewayRouteTable
	EC2: DeleteLocalGatewayRouteTableVirtualInterfaceGroupAssociation
	EC2: DeleteLocalGatewayRouteTableVpcAssociation
	EC2: DeleteManagedPrefixList
	EC2: DeleteNatGateway
	EC2: DeleteNetworkAcl
	EC2: DeleteNetworkAclEntry
	EC2: DeleteNetworkInsightsAccessScope

Awalan layanan	Tindakan
	EC2: DeleteNetworkInsightsAccessScopeAnalysis
	EC2: DeleteNetworkInsightsAnalysis
	EC2: DeleteNetworkInsightsPath
	EC2: DeleteNetworkInterface
	EC2: DeleteNetworkInterfacePermission
	EC2: DeletePlacementGroup
	ec2:4Kolam DeletePublicIpv
	EC2: DeleteQueuedReservedInstances
	EC2: DeleteRoute
	EC2: DeleteRouteTable
	EC2: DeleteSecurityGroup
	EC2: DeleteSnapshot
	EC2: DeleteSpotDatafeedSubscription
	EC2: DeleteSubnet
	EC2: DeleteSubnetCidrReservation
	EC2: DeleteTrafficMirrorFilter
	EC2: DeleteTrafficMirrorFilterRule
	EC2: DeleteTrafficMirrorSession
	EC2: DeleteTrafficMirrorTarget
	EC2: DeleteTransitGateway
	EC2: DeleteTransitGatewayConnect

Awalan layanan	Tindakan
	EC2: DeleteTransitGatewayConnectPeer
	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: DeleteVpcEndpointConnectionNotifications
	EC2: DeleteVpcEndpoints
	EC2: DeleteVpcEndpointServiceConfigurations
	EC2: DeleteVpcPeeringConnection
	EC2: DeleteVpnConnection
	EC2: DeleteVpnConnectionRoute

Awalan layanan	Tindakan
	EC2: DeleteVpnGateway
	EC2: DeprovisionByoipCidr
	EC2: DeprovisionIpamByoasn
	EC2: DeprovisionIpamPoolCidr
	ec2: DeprovisionPublicIpv4PoolCidr
	EC2: DeregisterImage
	EC2: DeregisterInstanceEventNotificationAttributes
	EC2: DeregisterTransitGatewayMulticastGroupMembers
	EC2: DeregisterTransitGatewayMulticastGroupSources
	EC2: DescribeAccountAttributes
	EC2: DescribeAddresses
	EC2: DescribeAddressesAttribute
	EC2: DescribeAddressTransfers
	EC2: DescribeAggregateIdFormat
	EC2: DescribeAvailabilityZones
	EC2: DescribeAwsNetworkPerformanceMetricSubscriptions
	EC2: DescribeBundleTasks
	EC2: DescribeByoipCidrs
	EC2: DescribeCapacityReservationFleets
	EC2: DescribeCapacityReservations
	EC2: DescribeCarrierGateways

Awalan layanan	Tindakan
	EC2: DescribeClassicLinkInstances
	EC2: DescribeClientVpnAuthorizationRules
	EC2: DescribeClientVpnConnections
	EC2: DescribeClientVpnEndpoints
	EC2: DescribeClientVpnRoutes
	EC2: DescribeClientVpnTargetNetworks
	EC2: DescribeCoipPools
	EC2: DescribeConversionTasks
	EC2: DescribeCustomerGateways
	EC2: DescribeDhcpOptions
	EC2: DescribeEgressOnlyInternetGateways
	EC2: DescribeElasticGpus
	EC2: DescribeExportImageTasks
	EC2: DescribeExportTasks
	EC2: DescribeFastLaunchImages
	EC2: DescribeFastSnapshotRestores
	EC2: DescribeFleetHistory
	EC2: DescribeFleetInstances
	EC2: DescribeFleets
	EC2: DescribeFlowLogs
	EC2: DescribeFpgaImageAttribute

Awalan layanan	Tindakan
	EC2: DescribeFpgaImages
	EC2: DescribeHostReservationOfferings
	EC2: DescribeHostReservations
	EC2: DescribeHosts
	EC2: DescribeIamInstanceProfileAssociations
	EC2: DescribeIdentityIdFormat
	EC2: DescribeIdFormat
	EC2: DescribeImageAttribute
	EC2: DescribeImages
	EC2: DescribeImportImageTasks
	EC2: DescribeImportSnapshotTasks
	EC2: DescribeInstanceAttribute
	EC2: DescribeInstanceConnectEndpoints
	EC2: DescribeInstanceCreditSpecifications
	EC2: DescribeInstanceEventNotificationAttributes
	EC2: DescribeInstanceEventWindows
	EC2: DescribeInstances
	EC2: DescribeInstanceStatus
	EC2: DescribeInstanceTopology
	EC2: DescribeInstanceTypeOfferings
	EC2: DescribeInstanceTypes

Awalan layanan	Tindakan
	EC2: DescribeInternetGateways
	EC2: DescribePamByoasn
	EC2: DescribePamPools
	EC2: DescribePamResourceDiscoveries
	EC2: DescribePamResourceDiscoveryAssociations
	EC2: DescribePams
	EC2: DescribePamScopes
	ec2:6Kolam DescribePv
	EC2: DescribeKeyPairs
	EC2: DescribeLaunchTemplates
	EC2: DescribeLaunchTemplateVersions
	EC2: DescribeLocalGatewayRouteTables
	EC2: DescribeLocalGatewayRouteTableVirtualInterfaceGroupAssociations
	EC2: DescribeLocalGatewayRouteTableVpcAssociations
	EC2: DescribeLocalGateways
	EC2: DescribeLocalGatewayVirtualInterfaceGroups
	EC2: DescribeLocalGatewayVirtualInterfaces
	EC2: DescribeLockedSnapshots
	EC2: DescribeMacHosts
	EC2: DescribeManagedPrefixLists

Awalan layanan	Tindakan
	EC2: DescribeMovingAddresses
	EC2: DescribeNatGateways
	EC2: DescribeNetworkAcls
	EC2: DescribeNetworkInsightsAccessScopeAnalyses
	EC2: DescribeNetworkInsightsAccessScopes
	EC2: DescribeNetworkInsightsAnalyses
	EC2: DescribeNetworkInsightsPaths
	EC2: DescribeNetworkInterfaceAttribute
	EC2: DescribeNetworkInterfacePermissions
	EC2: DescribeNetworkInterfaces
	EC2: DescribePlacementGroups
	EC2: DescribePrefixLists
	EC2: DescribePrincipalIdFormat
	ec2:4Pools DescribePublicIpv
	EC2: DescribeRegions
	EC2: DescribeReplaceRootVolumeTasks
	EC2: DescribeReservedInstances
	EC2: DescribeReservedInstancesListings
	EC2: DescribeReservedInstancesModifications
	EC2: DescribeReservedInstancesOfferings
	EC2: DescribeRouteTables

Awalan layanan	Tindakan
	EC2: DescribeScheduledInstanceAvailability
	EC2: DescribeScheduledInstances
	EC2: DescribeSecurityGroupReferences
	EC2: DescribeSecurityGroupRules
	EC2: DescribeSecurityGroups
	EC2: DescribeSnapshotAttribute
	EC2: DescribeSnapshots
	EC2: DescribeSnapshotTierStatus
	EC2: DescribeSpotDatafeedSubscription
	EC2: DescribeSpotFleetInstances
	EC2: DescribeSpotFleetRequestHistory
	EC2: DescribeSpotFleetRequests
	EC2: DescribeSpotInstanceRequests
	EC2: DescribeSpotPriceHistory
	EC2: DescribeStaleSecurityGroups
	EC2: DescribeStoreImageTasks
	EC2: DescribeSubnets
	EC2: DescribeTrafficMirrorFilters
	EC2: DescribeTrafficMirrorSessions
	EC2: DescribeTrafficMirrorTargets
	EC2: DescribeTransitGatewayAttachments

Awalan layanan	Tindakan
	EC2: DescribeTransitGatewayConnectPeers
	EC2: DescribeTransitGatewayConnects
	EC2: DescribeTransitGatewayMulticastDomains
	EC2: DescribeTransitGatewayPeeringAttachments
	EC2: DescribeTransitGatewayPolicyTables
	EC2: DescribeTransitGatewayRouteTableAnnouncements
	EC2: DescribeTransitGatewayRouteTables
	EC2: DescribeTransitGateways
	EC2: DescribeTransitGatewayVpcAttachments
	EC2: DescribeTrunkInterfaceAssociations
	EC2: DescribeVerifiedAccessEndpoints
	EC2: DescribeVerifiedAccessGroups
	EC2: DescribeVerifiedAccessInstanceLoggingConfigurations
	EC2: DescribeVerifiedAccessInstances
	EC2: DescribeVerifiedAccessTrustProviders
	EC2: DescribeVolumeAttribute
	EC2: DescribeVolumes
	EC2: DescribeVolumesModifications
	EC2: DescribeVolumeStatus
	EC2: DescribeVpcAttribute
	EC2: DescribeVpcClassicLink

Awalan layanan	Tindakan
	EC2: DescribeVpcClassicLinkDnsSupport
	EC2: DescribeVpcEndpointConnectionNotifications
	EC2: DescribeVpcEndpointConnections
	EC2: DescribeVpcEndpoints
	EC2: DescribeVpcEndpointServiceConfigurations
	EC2: DescribeVpcEndpointServicePermissions
	EC2: DescribeVpcEndpointServices
	EC2: DescribeVpcPeeringConnections
	EC2: DescribeVpcs
	EC2: DescribeVpnConnections
	EC2: DescribeVpnGateways
	EC2: DetachClassicLinkVpc
	EC2: DetachInternetGateway
	EC2: DetachNetworkInterface
	EC2: DetachVerifiedAccessTrustProvider
	EC2: DetachVolume
	EC2: DetachVpnGateway
	EC2: DisableAddressTransfer
	EC2: DisableAwsNetworkPerformanceMetricSubscription
	EC2: DisableEbsEncryptionByDefault
	EC2: DisableFastLaunch

Awalan layanan	Tindakan
	EC2: DisableFastSnapshotRestores
	EC2: DisableImage
	EC2: DisableImageBlockPublicAccess
	EC2: DisableImageDeprecation
	EC2: DisableIamOrganizationAdminAccount
	EC2: DisableSerialConsoleAccess
	EC2: DisableSnapshotBlockPublicAccess
	EC2: DisableTransitGatewayRouteTablePropagation
	EC2: DisableVgwRoutePropagation
	EC2: DisableVpcClassicLink
	EC2: DisableVpcClassicLinkDnsSupport
	EC2: DisassociateAddress
	EC2: DisassociateClientVpnTargetNetwork
	EC2: DisassociateEnclaveCertificateIamRole
	EC2: DisassociateIamInstanceProfile
	EC2: DisassociateInstanceEventWindow
	EC2: DisassociateIamByoasn
	EC2: DisassociateIamResourceDiscovery
	EC2: DisassociateNatGatewayAddress
	EC2: DisassociateRouteTable
	EC2: DisassociateSubnetCidrBlock

Awalan layanan	Tindakan
	EC2: DisassociateTransitGatewayMulticastDomain
	EC2: DisassociateTransitGatewayPolicyTable
	EC2: DisassociateTransitGatewayRouteTable
	EC2: DisassociateTrunkInterface
	EC2: DisassociateVpcCidrBlock
	EC2: EnableAddressTransfer
	EC2: EnableAwsNetworkPerformanceMetricSubscription
	EC2: EnableEbsEncryptionByDefault
	EC2: EnableFastLaunch
	EC2: EnableFastSnapshotRestores
	EC2: EnableImage
	EC2: EnableImageBlockPublicAccess
	EC2: EnableImageDeprecation
	EC2: EnableIpamOrganizationAdminAccount
	EC2: EnableReachabilityAnalyzerOrganizationSharing
	EC2: EnableSerialConsoleAccess
	EC2: EnableSnapshotBlockPublicAccess
	EC2: EnableTransitGatewayRouteTablePropagation
	EC2: EnableVgwRoutePropagation
	ec2: EnableVolume IO
	EC2: EnableVpcClassicLink

Awalan layanan	Tindakan
	EC2: EnableVpcClassicLinkDnsSupport
	EC2: ExportClientVpnClientCertificateRevocationList
	EC2: ExportClientVpnClientConfiguration
	EC2: ExportImage
	EC2: ExportTransitGatewayRoutes
	EC2: GetAssociatedEnclaveCertificateIamRoles
	ec2:6 GetAssociatedIpv PoolCidrs
	EC2: GetAwsNetworkPerformanceData
	EC2: GetCapacityReservationUsage
	EC2: GetCoipPoolUsage
	EC2: GetConsoleOutput
	EC2: GetConsoleScreenshot
	EC2: GetDefaultCreditSpecification
	EC2: GetEbsDefaultKmsKeyId
	EC2: GetEbsEncryptionByDefault
	EC2: GetFlowLogsIntegrationTemplate
	EC2: GetGroupsForCapacityReservation
	EC2: GetHostReservationPurchasePreview
	EC2: GetImageBlockPublicAccessState
	EC2: GetInstanceMetadataDefaults
	EC2: GetInstanceTypesFromInstanceRequirements

Awalan layanan	Tindakan
	EC2: GetInstanceUefiData
	EC2: GetIpamAddressHistory
	EC2: GetIpamDiscoveredAccounts
	EC2: GetIpamDiscoveredPublicAddresses
	EC2: GetIpamDiscoveredResourceCidrs
	EC2: GetIpamPoolAllocations
	EC2: GetIpamPoolCidrs
	EC2: GetIpamResourceCidrs
	EC2: GetLaunchTemplateData
	EC2: GetManagedPrefixListAssociations
	EC2: GetManagedPrefixListEntries
	EC2: GetNetworkInsightsAccessScopeAnalysisFindings
	EC2: GetNetworkInsightsAccessScopeContent
	EC2: GetPasswordData
	EC2: GetReservedInstancesExchangeQuote
	EC2: GetSecurityGroupsForVpc
	EC2: GetSerialConsoleAccessStatus
	EC2: GetSnapshotBlockPublicAccessState
	EC2: GetSpotPlacementScores
	EC2: GetSubnetCidrReservations
	EC2: GetTransitGatewayAttachmentPropagations

Awalan layanan	Tindakan
	EC2: GetTransitGatewayMulticastDomainAssociations
	EC2: GetTransitGatewayPolicyTableAssociations
	EC2: GetTransitGatewayPolicyTableEntries
	EC2: GetTransitGatewayPrefixListReferences
	EC2: GetTransitGatewayRouteTableAssociations
	EC2: GetTransitGatewayRouteTablePropagations
	EC2: GetVerifiedAccessEndpointPolicy
	EC2: GetVerifiedAccessGroupPolicy
	EC2: GetVpnConnectionDeviceSampleConfiguration
	EC2: GetVpnConnectionDeviceTypes
	EC2: GetVpnTunnelReplacementStatus
	EC2: ImportClientVpnClientCertificateRevocationList
	EC2: ImportImage
	EC2: ImportInstance
	EC2: ImportKeyPair
	EC2: ImportSnapshot
	EC2: ImportVolume
	EC2: ListImagesInRecycleBin
	EC2: ListSnapshotsInRecycleBin
	EC2: LockSnapshot
	EC2: ModifyAddressAttribute

Awalan layanan	Tindakan
	EC2: ModifyAvailabilityZoneGroup
	EC2: ModifyCapacityReservation
	EC2: ModifyCapacityReservationFleet
	EC2: ModifyClientVpnEndpoint
	EC2: ModifyDefaultCreditSpecification
	EC2: ModifyEbsDefaultKmsKeyId
	EC2: ModifyFleet
	EC2: ModifyFpgaImageAttribute
	EC2: ModifyHosts
	EC2: ModifyIdentityIdFormat
	EC2: ModifyIdFormat
	EC2: ModifyImageAttribute
	EC2: ModifyInstanceAttribute
	EC2: ModifyInstanceCapacityReservationAttributes
	EC2: ModifyInstanceCreditSpecification
	EC2: ModifyInstanceEventStartTime
	EC2: ModifyInstanceEventWindow
	EC2: ModifyInstanceMaintenanceOptions
	EC2: ModifyInstanceMetadataDefaults
	EC2: ModifyInstanceMetadataOptions
	EC2: ModifyInstancePlacement

Awalan layanan	Tindakan
	EC2: ModifyIpam
	EC2: ModifyIpamPool
	EC2: ModifyIpamResourceCidr
	EC2: ModifyIpamResourceDiscovery
	EC2: ModifyIpamScope
	EC2: ModifyLaunchTemplate
	EC2: ModifyLocalGatewayRoute
	EC2: ModifyManagedPrefixList
	EC2: ModifyNetworkInterfaceAttribute
	EC2: ModifyPrivateDnsNameOptions
	EC2: ModifyReservedInstances
	EC2: ModifySecurityGroupRules
	EC2: ModifySnapshotAttribute
	EC2: ModifySnapshotTier
	EC2: ModifySpotFleetRequest
	EC2: ModifySubnetAttribute
	EC2: ModifyTrafficMirrorFilterNetworkServices
	EC2: ModifyTrafficMirrorFilterRule
	EC2: ModifyTrafficMirrorSession
	EC2: ModifyTransitGateway
	EC2: ModifyTransitGatewayPrefixListReference

Awalan layanan	Tindakan
	EC2: ModifyTransitGatewayVpcAttachment
	EC2: ModifyVerifiedAccessEndpoint
	EC2: ModifyVerifiedAccessEndpointPolicy
	EC2: ModifyVerifiedAccessGroup
	EC2: ModifyVerifiedAccessGroupPolicy
	EC2: ModifyVerifiedAccessInstance
	EC2: ModifyVerifiedAccessInstanceLoggingConfiguration
	EC2: ModifyVerifiedAccessTrustProvider
	EC2: ModifyVolume
	EC2: ModifyVolumeAttribute
	EC2: ModifyVpcAttribute
	EC2: ModifyVpcEndpoint
	EC2: ModifyVpcEndpointConnectionNotification
	EC2: ModifyVpcEndpointServiceConfiguration
	EC2: ModifyVpcEndpointServicePayerResponsibility
	EC2: ModifyVpcEndpointServicePermissions
	EC2: ModifyVpcPeeringConnectionOptions
	EC2: ModifyVpcTenancy
	EC2: ModifyVpnConnection
	EC2: ModifyVpnConnectionOptions
	EC2: ModifyVpnTunnelCertificate

Awalan layanan	Tindakan
	EC2: ModifyVpnTunnelOptions
	EC2: MonitorInstances
	EC2: MoveAddressToVpc
	EC2: MoveByoipCidrToIpam
	EC2: ProvisionByoipCidr
	EC2: ProvisionIpamByoasn
	EC2: ProvisionIpamPoolCidr
	ec2: ProvisionPublicIpv4PoolCidr
	EC2: PurchaseHostReservation
	EC2: PurchaseReservedInstancesOffering
	EC2: PurchaseScheduledInstances
	EC2: RebootInstances
	EC2: RegisterImage
	EC2: RegisterInstanceEventNotificationAttributes
	EC2: RegisterTransitGatewayMulticastGroupMembers
	EC2: RegisterTransitGatewayMulticastGroupSources
	EC2: RejectTransitGatewayMulticastDomainAssociations
	EC2: RejectTransitGatewayPeeringAttachment
	EC2: RejectTransitGatewayVpcAttachment
	EC2: RejectVpcEndpointConnections
	EC2: RejectVpcPeeringConnection

Awalan layanan	Tindakan
	EC2: ReleaseAddress
	EC2: ReleaseHosts
	EC2: ReleaseIpamPoolAllocation
	EC2: ReplaceIamInstanceProfileAssociation
	EC2: ReplaceNetworkAclAssociation
	EC2: ReplaceNetworkAclEntry
	EC2: ReplaceRoute
	EC2: ReplaceRouteTableAssociation
	EC2: ReplaceTransitGatewayRoute
	EC2: ReplaceVpnTunnel
	EC2: ReportInstanceStatus
	EC2: RequestSpotFleet
	EC2: RequestSpotInstances
	EC2: ResetAddressAttribute
	EC2: ResetEbsDefaultKmsKeyId
	EC2: ResetFpgaImageAttribute
	EC2: ResetImageAttribute
	EC2: ResetInstanceAttribute
	EC2: ResetNetworkInterfaceAttribute
	EC2: ResetSnapshotAttribute
	EC2: RestoreAddressToClassic

Awalan layanan	Tindakan
	EC2: RestoreImageFromRecycleBin
	EC2: RestoreManagedPrefixListVersion
	EC2: RestoreSnapshotFromRecycleBin
	EC2: RestoreSnapshotTier
	EC2: RevokeClientVpnIngress
	EC2: RevokeSecurityGroupEgress
	EC2: RevokeSecurityGroupIngress
	EC2: RunInstances
	EC2: RunScheduledInstances
	EC2: SearchLocalGatewayRoutes
	EC2: SearchTransitGatewayMulticastGroups
	EC2: SearchTransitGatewayRoutes
	EC2: SendDiagnosticInterrupt
	EC2: StartInstances
	EC2: StartNetworkInsightsAccessScopeAnalysis
	EC2: StartNetworkInsightsAnalysis
	EC2: StartVpcEndpointServicePrivateDnsVerification
	EC2: StopInstances
	EC2: TerminateClientVpnConnections
	EC2: TerminateInstances
	ec2:6Alamat UnassignIpv

Awalan layanan	Tindakan
	EC2: UnassignPrivateIpAddresses EC2: UnassignPrivateNatGatewayAddress EC2: UnlockSnapshot EC2: UnmonitorInstances EC2: UpdateSecurityGroupRuleDescriptionsEgress EC2: UpdateSecurityGroupRuleDescriptionsIngress EC2: WithdrawByoipCidr

Awalan layanan	Tindakan
ecr	ecr: BatchCheckLayerAvailability ecr: BatchDeleteImage ecr: BatchGetImage ecr: BatchGetRepositoryScanningConfiguration ecr: CompleteLayerUpload ecr: CreatePullThroughCacheRule ecr: CreateRepository ecr: CreateRepositoryCreationTemplate ecr: DeleteLifecyclePolicy ecr: DeletePullThroughCacheRule ecr: DeleteRegistryPolicy ecr: DeleteRepository ecr: DeleteRepositoryCreationTemplate ecr: DeleteRepositoryPolicy ecr: DescribeImageReplicationStatus ecr: DescribeImages ecr: DescribeImageScanFindings ecr: DescribePullThroughCacheRules ecr: DescribeRegistry ecr: DescribeRepositories ecr: GetAuthorizationToken

Awalan layanan	Tindakan
	<p>ecr: GetDownloadUriForLayer</p> <p>ecr: GetLifecyclePolicy</p> <p>ecr: GetLifecyclePolicyPreview</p> <p>ecr: GetRegistryPolicy</p> <p>ecr: GetRegistryScanningConfiguration</p> <p>ecr: GetRepositoryPolicy</p> <p>ecr: InitiateLayerUpload</p> <p>ecr: ListImages</p> <p>ecr: PutImage</p> <p>ecr: PutImageScanningConfiguration</p> <p>ecr: PutRegistryPolicy</p> <p>ecr: PutRegistryScanningConfiguration</p> <p>ecr: PutReplicationConfiguration</p> <p>ecr: StartImageScan</p> <p>ecr: StartLifecyclePolicyPreview</p> <p>ecr: UpdatePullThroughCacheRule</p> <p>ecr: UploadLayerPart</p> <p>ecr: ValidatePullThroughCacheRule</p>

Awalan layanan	Tindakan
ecr-publik	ecr-publik: BatchCheckLayerAvailability ecr-publik: BatchDeletelImage 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: CreateCapacityProvider ecs: CreateCluster ecs: CreateService ecs: CreateTaskSet ecs: DeleteAccountSetting ecs: DeleteAttributes ecs: DeleteCapacityProvider ecs: DeleteCluster ecs: DeleteService ecs: DeleteTaskDefinitions ecs: DeleteTaskSet ecs: DeregisterContainerInstance ecs: DeregisterTaskDefinition ecs: DescribeCapacityProviders ecs: DescribeClusters ecs: DescribeContainerInstances ecs: DescribeServices ecs: DescribeTaskDefinition ecs: DescribeTasks ecs: DescribeTaskSets ecs: DiscoverPollEndpoint

Awalan layanan	Tindakan
	ecs: ExecuteCommand
	ecs: GetTaskProtection
	ecs: ListAccountSettings
	ecs: ListAttributes
	ecs: ListClusters
	ecs: ListContainerInstances
	ecs: ListServices
	ecs: ListServicesByNamespace
	ecs: ListTaskDefinitionFamilies
	ecs: ListTaskDefinitions
	ecs: ListTasks
	ecs: PutAccountSetting
	ecs: PutAccountSettingDefault
	ecs: PutAttributes
	ecs: PutClusterCapacityProviders
	ecs: RegisterContainerInstance
	ecs: RegisterTaskDefinition
	ecs: RunTask
	ecs: StartTask
	ecs: StopTask
	ecs: SubmitAttachmentStateChanges

Awalan layanan	Tindakan
	ecs: SubmitContainerStateChange
	ecs: SubmitTaskStateChange
	ecs: UpdateCapacityProvider
	ecs: UpdateCluster
	ecs: UpdateClusterSettings
	ecs: UpdateContainerAgent
	ecs: UpdateContainerInstancesState
	ecs: UpdateService
	ecs: UpdateServicePrimaryTaskSet
	ecs: UpdateTaskProtection
	ecs: UpdateTaskSet

Awalan layanan	Tindakan
eks	eks: AssociateAccessPolicy eks: AssociateEncryptionConfig eks: AssociateIdentityProviderConfig eks: CreateAccessEntry eks: CreateAddon eks: CreateCluster eks: CreateEksAnywhereSubscription eks: CreateFargateProfile eks: CreateNodegroup eks: DeleteAccessEntry eks: DeleteAddon eks: DeleteCluster eks: DeleteEksAnywhereSubscription eks: DeleteFargateProfile eks: DeleteNodegroup eks: DeletePodIdentityAssociation eks: DeregisterCluster eks: DescribeAccessEntry eks: DescribeAddon eks: DescribeAddonConfiguration eks: DescribeAddonVersions

Awalan layanan	Tindakan
	eks: DescribeCluster
	eks: DescribeEksAnywhereSubscription
	eks: DescribeFargateProfile
	eks: DescribeIdentityProviderConfig
	eks: DescribeInsight
	eks: DescribeNodegroup
	eks: DescribePodIdentityAssociation
	eks: DescribeUpdate
	eks: DisassociateAccessPolicy
	eks: DisassociateIdentityProviderConfig
	eks: ListAccessEntries
	eks: ListAccessPolicies
	eks: ListAddons
	eks: ListAssociatedAccessPolicies
	eks: ListClusters
	eks: ListEksAnywhereSubscriptions
	eks: ListFargateProfiles
	eks: ListIdentityProviderConfigs
	eks: ListInsights
	eks: ListNodegroups
	eks: ListPodIdentityAssociations

Awalan layanan	Tindakan
	<p>eks: ListUpdates</p> <p>eks: RegisterCluster</p> <p>eks: UpdateAccessEntry</p> <p>eks: UpdateAddon</p> <p>eks: UpdateClusterConfig</p> <p>eks: UpdateClusterVersion</p> <p>eks: UpdateEksAnywhereSubscription</p> <p>eks: UpdateNodegroupConfig</p> <p>eks: UpdateNodegroupVersion</p> <p>eks: UpdatePodIdentityAssociation</p>
inferensi elastis	<p>inferensi elastis: DescribeAcceleratorOfferings</p> <p>inferensi elastis: DescribeAccelerators</p> <p>inferensi elastis: DescribeAcceleratorTypes</p>

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
	elasticache: DescribeServerlessCaches
	elasticache: DescribeServerlessCacheSnapshots
	elasticache: DescribeServiceUpdates
	elasticache: DescribeSnapshots
	elasticache: DescribeUpdateActions
	elasticache: DescribeUserGroups
	elasticache: DescribeUsers
	elasticache: DisassociateGlobalReplicationGroup
	elasticache: ExportServerlessCacheSnapshot
	elasticache: FailoverGlobalReplicationGroup
	elasticache: IncreaseNodeGroupsInGlobalReplicationGroup
	elasticache: IncreaseReplicaCount
	elasticache: ListAllowedNodeTypeModifications
	elasticache: ModifyCacheCluster
	elasticache: ModifyCacheParameterGroup
	elasticache: ModifyCacheSubnetGroup
	elasticache: ModifyGlobalReplicationGroup
	elasticache: ModifyReplicationGroup
	elasticache: ModifyReplicationGroupShardConfiguration
	elasticache: ModifyServerlessCache
	elasticache: 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: DescribeApplications tangkai elastis: DescribeApplicationVersions tangkai elastis: DescribeConfigurationOptions tangkai elastis: DescribeConfigurationSettings

Awalan layanan	Tindakan
	<p>tangkai elastis: DescribeEnvironmentHealth</p> <p>tangkai elastis: DescribeEnvironmentManagedActionHistory</p> <p>tangkai elastis: DescribeEnvironmentManagedActions</p> <p>tangkai elastis: DescribeEnvironmentResources</p> <p>tangkai elastis: DescribeEnvironments</p> <p>tangkai elastis: DescribeEvents</p> <p>tangkai elastis: DescribeInstancesHealth</p> <p>tangkai elastis: DescribePlatformVersion</p> <p>tangkai elastis: DisassociateEnvironmentOperationsRole</p> <p>tangkai elastis: ListAvailableSolutionStacks</p> <p>tangkai elastis: ListPlatformBranches</p> <p>tangkai elastis: ListPlatformVersions</p> <p>tangkai elastis: RebuildEnvironment</p> <p>tangkai elastis: RequestEnvironmentInfo</p> <p>tangkai elastis: RestartAppServer</p> <p>tangkai elastis: RetrieveEnvironmentInfo</p> <p>elasticbeanstalk: CNames SwapEnvironment</p> <p>tangkai elastis: TerminateEnvironment</p> <p>tangkai elastis: UpdateApplication</p> <p>tangkai elastis: UpdateApplicationResourceLifecycle</p> <p>tangkai elastis: UpdateApplicationVersion</p>

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: DescribeMountTargets sistem file elastis: DescribeMountTargetSecurityGroups 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:CreateLB CookieStickinessPolicy 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: DeleteTargetGroup elasticloadbalancing: DeleteTrustStore

Awalan layanan	Tindakan
	<p>elasticloadbalancing: DeregisterInstancesFromLoadBalancer</p> <p>elasticloadbalancing: DeregisterTargets</p> <p>elasticloadbalancing: DescribeAccountLimits</p> <p>elasticloadbalancing: DescribeInstanceHealth</p> <p>elasticloadbalancing: DescribeListenerCertificates</p> <p>elasticloadbalancing: DescribeListeners</p> <p>elasticloadbalancing: DescribeLoadBalancerAttributes</p> <p>elasticloadbalancing: DescribeLoadBalancerPolicies</p> <p>elasticloadbalancing: DescribeLoadBalancerPolicyTypes</p> <p>elasticloadbalancing: DescribeLoadBalancers</p> <p>elasticloadbalancing: DescribeRules</p> <p>ElasticLoadBalancing: DescribessIPolicies</p> <p>elasticloadbalancing: DescribeTargetGroupAttributes</p> <p>elasticloadbalancing: DescribeTargetGroups</p> <p>elasticloadbalancing: DescribeTargetHealth</p> <p>elasticloadbalancing: DescribeTrustStoreAssociations</p> <p>elasticloadbalancing: DescribeTrustStoreRevocations</p> <p>elasticloadbalancing: DescribeTrustStores</p> <p>elasticloadbalancing: DetachLoadBalancerFromSubnets</p> <p>elasticloadbalancing: DisableAvailabilityZonesForLoadBalancer</p> <p>elasticloadbalancing: EnableAvailabilityZonesForLoadBalancer</p>

Awalan layanan	Tindakan
	<p>elasticloadbalancing: GetTrustStoreCaCertificatesBundle</p> <p>elasticloadbalancing: GetTrustStoreRevocationContent</p> <p>elasticloadbalancing: ModifyListener</p> <p>elasticloadbalancing: ModifyLoadBalancerAttributes</p> <p>elasticloadbalancing: ModifyRule</p> <p>elasticloadbalancing: ModifyTargetGroup</p> <p>elasticloadbalancing: ModifyTargetGroupAttributes</p> <p>elasticloadbalancing: ModifyTrustStore</p> <p>elasticloadbalancing: RegisterInstancesWithLoadBalancer</p> <p>elasticloadbalancing: RegisterTargets</p> <p>elasticloadbalancing: RemoveListenerCertificates</p> <p>elasticloadbalancing: RemoveTrustStoreRevocations</p> <p>elasticloadbalancing: SetIpAddressType</p> <p>elasticloadbalancing: SSLCertificate SetLoadBalancerListener</p> <p>elasticloadbalancing: SetLoadBalancerPoliciesForBackendServer</p> <p>elasticloadbalancing: SetLoadBalancerPoliciesOfListener</p> <p>elasticloadbalancing: SetRulePriorities</p> <p>elasticloadbalancing: SetSecurityGroups</p> <p>elasticloadbalancing: SetSubnets</p>

Awalan layanan	Tindakan
elastictranscoder	elastictranscoder: CancelJob elastictranscoder: CreateJob elastictranscoder: CreatePipeline elastictranscoder: CreatePreset elastictranscoder: DeletePipeline elastictranscoder: DeletePreset elastictranscoder: ListJobsByPipeline elastictranscoder: ListJobsByStatus elastictranscoder: ListPipelines elastictranscoder: ListPresets elastictranscoder: ReadJob elastictranscoder: ReadPipeline elastictranscoder: ReadPreset elastictranscoder: TestRole elastictranscoder: UpdatePipeline elastictranscoder: UpdatePipelineNotifications elastictranscoder: UpdatePipelineStatus

Awalan layanan	Tindakan
kontainer emr	wadah emr: CancelJobRun wadah emr: CreateJobTemplate wadah emr: CreateManagedEndpoint wadah emr: CreateVirtualCluster wadah emr: DeleteJobTemplate wadah emr: DeleteManagedEndpoint wadah emr: DeleteVirtualCluster wadah emr: DescribeJobRun wadah emr: DescribeJobTemplate wadah emr: DescribeManagedEndpoint wadah emr: DescribeVirtualCluster wadah emr: GetManagedEndpointSessionCredentials wadah emr: ListJobRuns wadah emr: ListJobTemplates wadah emr: ListManagedEndpoints wadah emr: ListVirtualClusters wadah emr: 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: ListApplications emr-tanpa server: ListJobRuns emr-tanpa server: StartApplication emr-tanpa server: StartJobRun emr-tanpa server: StopApplication 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: CreateOutboundConnection es: CreateOutboundCrossClusterSearchConnection es: CreatePackage es: CreateVpcEndpoint es: DeleteDomain es: DeleteElasticsearchDomain es: DeleteElasticsearchServiceRole es: DeleteInboundConnection es: DeleteInboundCrossClusterSearchConnection es: DeleteOutboundConnection es: DeleteOutboundCrossClusterSearchConnection es: DeletePackage es: DeleteVpcEndpoint

Awalan layanan	Tindakan
	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
	es: DescribeReservedElasticsearchInstances
	es: DescribeReservedInstanceOfferings

Awalan layanan	Tindakan
	<ul style="list-style-type: none"><li data-bbox="542 212 992 243">es: DescribeReservedInstances<li data-bbox="542 291 919 323">es: DescribeVpcEndpoints<li data-bbox="542 371 867 403">es: DissociatePackage<li data-bbox="542 451 1122 483">es: GetCompatibleElasticsearchVersions<li data-bbox="542 531 932 562">es: GetCompatibleVersions<li data-bbox="542 611 816 642">es: GetDataSource<li data-bbox="542 690 1029 722">es: GetDomainMaintenanceStatus<li data-bbox="542 770 979 802">es: GetPackageVersionHistory<li data-bbox="542 850 867 882">es: GetUpgradeHistory<li data-bbox="542 930 859 961">es: GetUpgradeStatus<li data-bbox="542 1010 829 1041">es: ListDataSources<li data-bbox="542 1089 857 1121">es: ListDomainNames<li data-bbox="542 1169 943 1201">es: ListDomainsForPackage<li data-bbox="542 1249 1040 1281">es: ListElasticsearchInstanceTypes<li data-bbox="542 1329 959 1360">es: ListElasticsearchVersions<li data-bbox="542 1409 935 1440">es: ListInstanceTypeDetails<li data-bbox="542 1488 943 1520">es: ListPackagesForDomain<li data-bbox="542 1568 899 1600">es: ListScheduledActions<li data-bbox="542 1648 769 1680">es: ListVersions<li data-bbox="542 1728 927 1759">es: ListVpcEndpointAccess<li data-bbox="542 1808 841 1839">es: ListVpcEndpoints

Awalan layanan	Tindakan
	<ul style="list-style-type: none"><li data-bbox="542 212 1000 247">es: ListVpcEndpointsForDomain<li data-bbox="542 296 1289 331">es: PurchaseReservedElasticsearchInstanceOffering<li data-bbox="542 380 1101 415">es: PurchaseReservedInstanceOffering<li data-bbox="542 464 964 499">es: RejectInboundConnection<li data-bbox="542 548 1247 583">es: RejectInboundCrossClusterSearchConnection<li data-bbox="542 632 987 667">es: RevokeVpcEndpointAccess<li data-bbox="542 716 954 751">es: StartDomainMaintenance<li data-bbox="542 800 1187 835">es: StartElasticsearchServiceSoftwareUpdate<li data-bbox="542 884 997 919">es: StartServiceSoftwareUpdate<li data-bbox="542 968 867 1003">es: UpdateDataSource<li data-bbox="542 1052 899 1087">es: UpdateDomainConfig<li data-bbox="542 1136 1089 1171">es: UpdateElasticsearchDomainConfig<li data-bbox="542 1220 824 1255">es: UpdatePackage<li data-bbox="542 1304 938 1339">es: UpdateScheduledAction<li data-bbox="542 1388 883 1423">es: UpdateVpcEndpoint<li data-bbox="542 1472 829 1507">es: UpgradeDomain<li data-bbox="542 1556 1019 1591">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

Awalan layanan	Tindakan
ternyata	jelas: CreateExperiment jelas: CreateFeature jelas: CreateLaunch jelas: CreateProject jelas: CreateSegment jelas: DeleteExperiment jelas: DeleteFeature jelas: DeleteLaunch jelas: DeleteProject jelas: DeleteSegment jelas: GetExperiment jelas: GetExperimentResults jelas: GetFeature jelas: GetLaunch jelas: GetProject jelas: GetSegment jelas: ListExperiments jelas: ListFeatures jelas: ListLaunches jelas: ListProjects jelas: ListSegmentReferences

Awalan layanan	Tindakan
	jelas: ListSegments jelas: StartExperiment jelas: StartLaunch jelas: StopExperiment jelas: StopLaunch jelas: TestSegmentPattern jelas: UpdateExperiment jelas: UpdateFeature jelas: UpdateLaunch jelas: UpdateProject jelas: UpdateProjectDataDelivery

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
	<p>ruang finspace: ListUsers</p> <p>ruang finspace: LoadSampleDataSetGroupIntoEnvironment</p> <p>ruang finspace: ResetUserPassword</p> <p>ruang finspace: UpdateEnvironment</p> <p>ruang finspace: UpdateKxClusterCodeConfiguration</p> <p>ruang finspace: UpdateKxClusterDatabases</p> <p>ruang finspace: UpdateKxDatabase</p> <p>ruang finspace: UpdateKxDataview</p> <p>ruang finspace: UpdateKxEnvironment</p> <p>ruang finspace: UpdateKxEnvironmentNetwork</p> <p>ruang finspace: UpdateKxUser</p> <p>ruang finspace: UpdateKxVolume</p> <p>ruang finspace: UpdateUser</p>
firehose	<p>selang api: CreateDeliveryStream</p> <p>selang api: DeleteDeliveryStream</p> <p>selang api: DescribeDeliveryStream</p> <p>selang api: ListDeliveryStreams</p> <p>selang api: StartDeliveryStreamEncryption</p> <p>selang api: StopDeliveryStreamEncryption</p> <p>selang api: UpdateDestination</p>

Awalan layanan	Tindakan
fis	fis: CreateExperimentTemplate fis: CreateTargetAccountConfiguration fis: DeleteExperimentTemplate fis: DeleteTargetAccountConfiguration fis: GetAction fis: GetExperiment fis: GetExperimentTargetAccountConfiguration fis: GetExperimentTemplate fis: GetTargetAccountConfiguration fis: GetTargetResourceType fis: ListActions fis: ListExperimentResolvedTargets fis: ListExperiments fis: ListExperimentTargetAccountConfigurations fis: ListExperimentTemplates fis: ListTargetAccountConfigurations fis: ListTargetResourceTypes fis: StartExperiment fis: StopExperiment fis: UpdateExperimentTemplate 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
	<p>fms: GetViolationDetails</p> <p>fms: ListAdminAccountsForOrganization</p> <p>fms: ListAdminsManagingAccount</p> <p>fms: ListAppsLists</p> <p>fms: ListComplianceStatus</p> <p>fms: ListDiscoveredResources</p> <p>fms: ListMemberAccounts</p> <p>fms: ListPolicies</p> <p>fms: ListProtocolsLists</p> <p>fms: ListResourceSetResources</p> <p>fms: ListResourceSets</p> <p>fms: ListThirdPartyFirewallFirewallPolicies</p> <p>fms: PutAdminAccount</p> <p>fms: PutAppsList</p> <p>fms: PutNotificationChannel</p> <p>fms: PutPolicy</p> <p>fms: PutProtocolsList</p> <p>fms: PutResourceSet</p>

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: DeleteEventsByEventType
	detektor penipuan: DeleteEventType
	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: GetDetectors
	detektor penipuan: GetDetectorVersion
	detektor penipuan: GetEntityTypes
	detektor penipuan: GetEvent
	detektor penipuan: GetEventPrediction
	detektor penipuan: GetEventPredictionMetadata
	detektor penipuan: GetEventTypes
	detektor penipuan: GetExternalModels
	Detektor Penipu: GetKMS EncryptionKey

Awalan layanan	Tindakan
	detektor penipuan: GetLabels
	detektor penipuan: GetListElements
	detektor penipuan: GetListsMetadata
	detektor penipuan: GetModels
	detektor penipuan: GetModelVersion
	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 Penipu:Putkms EncryptionKey
	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: DescribeSharedVpcConfiguration
	fsx: DescribeSnapshots
	fsx: DescribeStorageVirtualMachines
	fsx: DescribeVolumes
	fsx: DisassociateFileSystemAliases
	fsx: ReleaseFileSystemNfs V3Locks
	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: 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: DeleteFleet gamelift: DeleteFleetLocations

Awalan layanan	Tindakan
	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:DescribeEC2 InstanceLimits
	gamelift: DescribeFleetAttributes
	gamelift: DescribeFleetCapacity
	gamelift: DescribeFleetEvents
	gamelift: DescribeFleetLocationAttributes
	gamelift: DescribeFleetLocationCapacity
	gamelift: DescribeFleetLocationUtilization

Awalan layanan	Tindakan
	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
	gamelift: DescribeScript
	gamelift: DescribeVpcPeeringAuthorizations
	gamelift: DescribeVpcPeeringConnections
	gamelift: GetComputeAccess
	gamelift: GetComputeAuthToken

Awalan layanan	Tindakan
	gamelift: GetGameSessionLogUrl
	gamelift: GetInstanceAccess
	gamelift: ListAliases
	gamelift: ListBuilds
	gamelift: ListCompute
	gamelift: ListFleets
	gamelift: ListGameServerGroups
	gamelift: ListGameServers
	gamelift: ListLocations
	gamelift: ListScripts
	gamelift: PutScalingPolicy
	gamelift: RegisterCompute
	gamelift: RegisterGameServer
	gamelift: RequestUploadCredentials
	gamelift: ResolveAlias
	gamelift: ResumeGameServerGroup
	gamelift: SearchGameSessions
	gamelift: StartFleetActions
	gamelift: StartGameSessionPlacement
	gamelift: StartMatchBackfill
	gamelift: StartMatchmaking

Awalan layanan	Tindakan
	gamelift: StopFleetActions
	gamelift: StopGameSessionPlacement
	gamelift: StopMatchmaking
	gamelift: SuspendGameServerGroup
	gamelift: UpdateAlias
	gamelift: UpdateBuild
	gamelift: UpdateFleetAttributes
	gamelift: UpdateFleetCapacity
	gamelift: UpdateFleetPortSettings
	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: CreateGeofenceCollection geografis: CreateMap geografis: CreatePlaceIndex geografis: CreateRouteCalculator geografis: CreateTracker geografis: DeleteGeofenceCollection geografis: DeleteKey geografis: DeleteMap geografis: DeletePlaceIndex geografis: DeleteRouteCalculator geografis: DeleteTracker geografis: DescribeGeofenceCollection

Awalan layanan	Tindakan
	<p>geografis: DescribeKey</p> <p>geografis: DescribeMap</p> <p>geografis: DescribePlaceIndex</p> <p>geografis: DescribeRouteCalculator</p> <p>geografis: DescribeTracker</p> <p>geografis: DisassociateTrackerConsumer</p> <p>geografis: GetDevicePosition</p> <p>geografis: GetDevicePositionHistory</p> <p>geografis: GetGeofence</p> <p>geografis: GetMapGlyphs</p> <p>geografis: GetMapSprites</p> <p>geografis: GetMapStyleDescriptor</p> <p>geografis: GetMapTile</p> <p>geografis: GetPlace</p> <p>geografis: ListDevicePositions</p> <p>geografis: ListGeofenceCollections</p> <p>geografis: ListGeofences</p> <p>geografis: ListKeys</p> <p>geografis: ListMaps</p> <p>geografis: ListPlaceIndexes</p> <p>geografis: ListRouteCalculators</p>

Awalan layanan	Tindakan
	geografis: ListTrackerConsumers geografis: ListTrackers geografis: PutGeofence geografis: SearchPlaceIndexForPosition geografis: SearchPlaceIndexForSuggestions geografis: SearchPlaceIndexForText geografis: UpdateGeofenceCollection geografis: UpdateKey geografis: UpdateMap geografis: UpdatePlaceIndex geografis: UpdateRouteCalculator geografis: UpdateTracker

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: DeleteWorkspace grafana: DeleteWorkspaceApiKey grafana: DescribeWorkspace grafana: DescribeWorkspaceAuthentication grafana: DescribeWorkspaceConfiguration grafana: DisassociateLicense grafana: ListPermissions grafana: ListVersions grafana: ListWorkspaces grafana: UpdatePermissions grafana: UpdateWorkspace grafana: UpdateWorkspaceAuthentication 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: ListComponents
	greengrass: ListComponentVersions
	greengrass: ListConnectorDefinitions
	greengrass: ListConnectorDefinitionVersions
	greengrass: ListCoreDefinitions
	greengrass: ListCoreDefinitionVersions
	greengrass: ListCoreDevices
	greengrass: ListDeployments
	greengrass: ListDeviceDefinitions
	greengrass: ListDeviceDefinitionVersions
	greengrass: ListEffectiveDeployments
	greengrass: ListFunctionDefinitions
	greengrass: ListFunctionDefinitionVersions
	greengrass: ListGroupCertificateAuthorities

Awalan layanan	Tindakan
	greengrass: ListGroups
	greengrass: ListGroupVersions
	greengrass: ListInstalledComponents
	greengrass: ListLoggerDefinitions
	greengrass: ListLoggerDefinitionVersions
	greengrass: ListResourceDefinitions
	greengrass: ListResourceDefinitionVersions
	greengrass: ListSubscriptionDefinitions
	greengrass: ListSubscriptionDefinitionVersions
	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: CreateEphemeris stasiun tanah: CreateMissionProfile stasiun tanah: DeleteConfig stasiun tanah: DeleteDataflowEndpointGroup stasiun tanah: DeleteEphemeris stasiun tanah: DeleteMissionProfile stasiun tanah: DescribeContact stasiun tanah: DescribeEphemeris stasiun tanah: GetConfig stasiun tanah: GetDataflowEndpointGroup stasiun tanah: GetMinuteUsage stasiun tanah: GetMissionProfile stasiun tanah: GetSatellite stasiun tanah: ListConfigs stasiun tanah: ListContacts stasiun tanah: ListDataflowEndpointGroups stasiun tanah: ListEphemerides stasiun tanah: ListGroundStations

Awalan layanan	Tindakan
	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: CreateMembers tugas jaga: CreatePublishingDestination tugas jaga: CreateSampleFindings tugas jaga: CreateThreatIntelSet tugas jaga: DeclineInvitations tugas jaga: DeleteDetector tugas jaga: DeleteFilter tugas jaga: DeleteInvitations guardDuty:deleteIPset tugas jaga: DeleteMembers tugas jaga: DeletePublishingDestination tugas jaga: DeleteThreatIntelSet tugas jaga: DescribeMalwareScans tugas jaga: DescribeOrganizationConfiguration tugas jaga: DescribePublishingDestination

Awalan layanan	Tindakan
	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
	tugas jaga: GetInvitationsCount
	GuardDuty:GETIPSET
	tugas jaga: GetMalwareScanSettings
	tugas jaga: GetMasterAccount
	tugas jaga: GetMemberDetectors
	tugas jaga: GetMembers
	tugas jaga: GetOrganizationStatistics
	tugas jaga: GetRemainingFreeTrialDays
	tugas jaga: GetThreatIntelSet
	tugas jaga: GetUsageStatistics

Awalan layanan	Tindakan
	tugas jaga: InviteMembers
	tugas jaga: ListCoverage
	tugas jaga: ListDetectors
	tugas jaga: ListFilters
	tugas jaga: ListFindings
	tugas jaga: ListInvitations
	guardduty:Listipsets
	tugas jaga: ListMembers
	tugas jaga: ListOrganizationAdminAccounts
	tugas jaga: ListPublishingDestinations
	tugas jaga: ListThreatIntelSets
	tugas jaga: SendSecurityTelemetry
	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: UpdateMalwareScanSettings

Awalan layanan	Tindakan
	tugas jaga: UpdateMemberDetectors tugas jaga: UpdateOrganizationConfiguration tugas jaga: UpdatePublishingDestination tugas jaga: UpdateThreatIntelSet
healthlake	Healthlake:CreateFhirDataStore Healthlake: CreateResource HealthLake:DeleteFhirDataStore Healthlake: DeleteResource Healthlake:DescribefhirDataStore Healthlake:Deskripsifhir ExportJob Healthlake:Deskripsifhir ImportJob Healthlake: GetCapabilities HealthLake:ListfHirDataStores Healthlake:Listfhir ExportJobs Healthlake:Listfhir ImportJobs Healthlake: ReadResource Healthlake: SearchWithGet Healthlake: SearchWithPost Healthlake:Startfhir ExportJob Healthlake:Startfhir ImportJob 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	iam: AddClient ID ID ToOpen ConnectProvider saya: AddRoleToInstanceProfile saya: AddUserToGroup saya: AttachGroupPolicy saya: AttachRolePolicy saya: AttachUserPolicy saya: ChangePassword saya: CreateAccessKey saya: CreateAccountAlias saya: CreateGroup saya: CreateInstanceProfile saya: CreateLoginProfile iam: ID CreateOpen ConnectProvider saya: CreatePolicy saya: CreatePolicyVersion saya: CreateRole iam:CreateSamlProvider saya: CreateServiceLinkedRole saya: CreateServiceSpecificCredential saya: CreateUser saya: MFADEvice CreateVirtual

Awalan layanan	Tindakan
	<p>iam:Nonaktifkan FADEVICE</p> <p>saya: DeleteAccessKey</p> <p>saya: DeleteAccountAlias</p> <p>saya: DeleteAccountPasswordPolicy</p> <p>saya: DeleteCloudFrontPublicKey</p> <p>saya: DeleteGroup</p> <p>saya: DeleteGroupPolicy</p> <p>saya: DeleteInstanceProfile</p> <p>saya: DeleteLoginProfile</p> <p>iam: ID DeleteOpen ConnectProvider</p> <p>saya: DeletePolicy</p> <p>saya: DeletePolicyVersion</p> <p>saya: DeleteRole</p> <p>saya: DeleteRolePermissionsBoundary</p> <p>saya: DeleteRolePolicy</p> <p>iam:deleteSamlProvider</p> <p>saya: DeleteServerCertificate</p> <p>saya: DeleteServiceLinkedRole</p> <p>saya: DeleteServiceSpecificCredential</p> <p>saya: DeleteSigningCertificate</p> <p>iam:deletessh PublicKey</p>

Awalan layanan	Tindakan
	saya: DeleteUser
	saya: DeleteUserPermissionsBoundary
	saya: DeleteUserPolicy
	saya: MFADEVICE DeleteVirtual
	saya: DetachGroupPolicy
	saya: DetachRolePolicy
	saya: DetachUserPolicy
	iam:enableMfadevice
	saya: GenerateCredentialReport
	saya: GenerateOrganizationsAccessReport
	saya: GenerateServiceLastAccessedDetails
	saya: GetAccessKeyLastUsed
	saya: GetAccountAuthorizationDetails
	saya: GetAccountEmailAddress
	saya: GetAccountName
	saya: GetAccountPasswordPolicy
	saya: GetAccountSummary
	saya: GetCloudFrontPublicKey
	saya: GetContextKeysForCustomPolicy
	saya: GetContextKeysForPrincipalPolicy
	saya: GetCredentialReport

Awalan layanan	Tindakan
	saya: GetGroup
	saya: GetGroupPolicy
	saya: GetInstanceProfile
	saya: GetLoginProfile
	iam:GetMFADevice
	iam: ID GetOpen ConnectProvider
	saya: GetOrganizationsAccessReport
	saya: GetPolicy
	saya: GetPolicyVersion
	saya: GetRole
	saya: GetRolePolicy
	iam:GetSamlProvider
	saya: GetServerCertificate
	saya: GetServiceLastAccessedDetails
	saya: GetServiceLastAccessedDetailsWithEntities
	saya: GetServiceLinkedRoleDeletionStatus
	iam:GetSSH PublicKey
	saya: GetUser
	saya: GetUserPolicy
	saya: ListAccessKeys
	saya: ListAccountAliases

Awalan layanan	Tindakan
	<p>saya: ListAttachedGroupPolicies</p> <p>saya: ListAttachedRolePolicies</p> <p>saya: ListAttachedUserPolicies</p> <p>saya: ListCloudFrontPublicKeys</p> <p>saya: ListEntitiesForPolicy</p> <p>saya: ListGroupPolicies</p> <p>saya: ListGroups</p> <p>saya: ListGroupsForUser</p> <p>saya: ListInstanceProfiles</p> <p>saya: ListInstanceProfilesForRole</p> <p>iam:ListmFADevices</p> <p>iam: ID ListOpen ConnectProviders</p> <p>saya: ListPolicies</p> <p>saya: ListPoliciesGrantingServiceAccess</p> <p>saya: ListPolicyVersions</p> <p>saya: ListRolePolicies</p> <p>saya: ListRoles</p> <p>iam:ListSamlProviders</p> <p>saya: ListServerCertificates</p> <p>saya: ListServiceSpecificCredentials</p> <p>saya: ListSigningCertificates</p>

Awalan layanan	Tindakan
	<p>iam:ListSSH PublicKeys</p> <p>IAM:daftar RegionalEndpointsStatus</p> <p>saya: ListUserPolicies</p> <p>saya: ListUsers</p> <p>iam: ListVirtual mfaDevices</p> <p>saya: PutGroupPolicy</p> <p>saya: PutRolePermissionsBoundary</p> <p>saya: PutRolePolicy</p> <p>saya: PutUserPermissionsBoundary</p> <p>saya: PutUserPolicy</p> <p>iam: RemoveClient ID ID FromOpen ConnectProvider</p> <p>saya: RemoveRoleFromInstanceProfile</p> <p>saya: RemoveUserFromGroup</p> <p>saya: ResetServiceSpecificCredential</p> <p>iam:resyncmfadevice</p> <p>saya: SetDefaultPolicyVersion</p> <p>saya: SetSecurityTokenServicePreferences</p> <p>IAM: Setsts RegionalEndpointStatus</p> <p>saya: SimulateCustomPolicy</p> <p>saya: SimulatePrincipalPolicy</p> <p>saya: UpdateAccessKey</p>

Awalan layanan	Tindakan
	saya: UpdateAccountEmailAddress saya: UpdateAccountName saya: UpdateAccountPasswordPolicy saya: UpdateAssumeRolePolicy saya: UpdateCloudFrontPublicKey saya: UpdateGroup saya: UpdateLoginProfile iam: ID UpdateOpen ConnectProviderThumbprint saya: UpdateRole saya: UpdateRoleDescription iam:UpdateSamlProvider saya: UpdateServerCertificate saya: UpdateServiceSpecificCredential saya: UpdateSigningCertificate iam:UpdatessH PublicKey saya: UpdateUser saya: UploadCloudFrontPublicKey saya: UploadServerCertificate saya: UploadSigningCertificate iam:UnggahSSH PublicKey

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: GetComponentPolicy

Awalan layanan	Tindakan
	<p>imagebuilder: GetContainerRecipePolicy</p> <p>imagebuilder: GetImagePolicy</p> <p>imagebuilder: GetImageRecipePolicy</p> <p>imagebuilder: GetLifecycleExecution</p> <p>imagebuilder: GetLifecyclePolicy</p> <p>imagebuilder: GetWorkflowExecution</p> <p>imagebuilder: GetWorkflowStepExecution</p> <p>imagebuilder: ImportComponent</p> <p>imagebuilder: ImportVmlImage</p> <p>imagebuilder: ListComponentBuildVersions</p> <p>imagebuilder: ListComponents</p> <p>imagebuilder: ListContainerRecipes</p> <p>imagebuilder: ListDistributionConfigurations</p> <p>imagebuilder: ListImageBuildVersions</p> <p>imagebuilder: ListImagePackages</p> <p>imagebuilder: ListImagePipelineImages</p> <p>imagebuilder: ListImagePipelines</p> <p>imagebuilder: ListImageRecipes</p> <p>imagebuilder: ListImages</p> <p>imagebuilder: ListImageScanFindingAggregations</p> <p>imagebuilder: ListImageScanFindings</p>

Awalan layanan	Tindakan
	<p>imagebuilder: ListInfrastructureConfigurations</p> <p>imagebuilder: ListLifecycleExecutionResources</p> <p>imagebuilder: ListLifecycleExecutions</p> <p>imagebuilder: ListLifecyclePolicies</p> <p>imagebuilder: ListWaitingWorkflowSteps</p> <p>imagebuilder: ListWorkflowExecutions</p> <p>imagebuilder: ListWorkflows</p> <p>imagebuilder: ListWorkflowStepExecutions</p> <p>imagebuilder: PutComponentPolicy</p> <p>imagebuilder: PutContainerRecipePolicy</p> <p>imagebuilder: PutImagePolicy</p> <p>imagebuilder: PutImageRecipePolicy</p> <p>imagebuilder: SendWorkflowStepAction</p> <p>imagebuilder: StartImagePipelineExecution</p> <p>imagebuilder: StartResourceStateUpdate</p> <p>imagebuilder: UpdateDistributionConfiguration</p> <p>imagebuilder: UpdateImagePipeline</p> <p>imagebuilder: UpdateInfrastructureConfiguration</p>

Awalan layanan	Tindakan
inspektor	inspektor: AddAttributesToFindings inspektor: CreateAssessmentTarget inspektor: CreateAssessmentTemplate inspektor: CreateExclusionsPreview inspektor: CreateResourceGroup inspektor: DeleteAssessmentRun inspektor: DeleteAssessmentTarget inspektor: DeleteAssessmentTemplate inspektor: DescribeAssessmentRuns inspektor: DescribeAssessmentTargets inspektor: DescribeAssessmentTemplates inspektor: DescribeCrossAccountAccessRole inspektor: DescribeExclusions inspektor: DescribeFindings inspektor: DescribeResourceGroups inspektor: DescribeRulesPackages inspektor: GetAssessmentReport inspektor: GetExclusionsPreview inspektor: GetTelemetryMetadata inspektor: ListAssessmentRunAgents inspektor: ListAssessmentRuns

Awalan layanan	Tindakan
	inspektor: ListAssessmentTargets
	inspektor: ListAssessmentTemplates
	inspektor: ListEventSubscriptions
	inspektor: ListExclusions
	inspektor: ListFindings
	inspektor: ListRulesPackages
	inspektor: PreviewAgents
	inspektor: RegisterCrossAccountAccessRole
	inspektor: RemoveAttributesFromFindings
	inspektor: StartAssessmentRun
	inspektor: StopAssessmentRun
	inspektor: SubscribeToEvent
	inspektor: UnsubscribeFromEvent
	inspektor: UpdateAssessmentTarget

Awalan layanan	Tindakan
inspektor2	inspektor2: AssociateMember inspektor2: BatchGetAccountStatus inspektor2: BatchGetCodeSnippet inspektor2: BatchGetFindingDetails inspektor2: BatchGetFreeTrialInfo inspektor2:2 BatchGetMemberEc DeepInspectionStatus inspektor2:2 BatchUpdateMemberEc DeepInspectionStatus inspektor2: CancelFindingsReport inspektor2: CancelSbomExport inspektor2: CreateCisScanConfiguration inspektor2: CreateFilter inspektor2: CreateFindingsReport inspektor2: CreateSbomExport inspektor2: DeleteCisScanConfiguration inspektor2: DeleteFilter inspektor2: DescribeOrganizationConfiguration Inspector2: Nonaktifkan inspektor2: DisableDelegatedAdminAccount inspektor2: DisassociateMember Inspector2: aktifkan inspektor2: EnableDelegatedAdminAccount

Awalan layanan	Tindakan
	inspektor2: GetCisScanReport
	inspektor2: GetCisScanResultDetails
	inspektor2: GetConfiguration
	inspektor2: GetDelegatedAdminAccount
	inspektor2:2 GetEc DeepInspectionConfiguration
	inspektor2: GetEncryptionKey
	inspektor2: GetFindingsReportStatus
	inspektor2: GetMember
	inspektor2: GetSbomExport
	inspektor2: ListAccountPermissions
	inspektor2: ListCisScanConfigurations
	inspektor2: ListCisScanResultsAggregatedByChecks
	inspektor2: ListCisScanResultsAggregatedByTargetResource
	inspektor2: ListCisScans
	inspektor2: ListCoverage
	inspektor2: ListCoverageStatistics
	inspektor2: ListDelegatedAdminAccounts
	inspektor2: ListFilters
	inspektor2: ListFindingAggregations
	inspektor2: ListFindings
	inspektor2: ListMembers

Awalan layanan	Tindakan
	inspektor2: ListUsageTotals inspektor2: ResetEncryptionKey inspektor2: SearchVulnerabilities inspektor2: SendCisSessionHealth inspektor2: SendCisSessionTelemetry inspektor2: StartCisSession inspektor2: StopCisSession inspektor2: UpdateCisScanConfiguration inspektor2: UpdateConfiguration inspektor2:2 UpdateEc DeepInspectionConfiguration inspektor2: UpdateEncryptionKey inspektor2: UpdateFilter inspektor2: UpdateOrganizationConfiguration inspektor2:2 UpdateOrgEc DeepInspectionConfiguration

Awalan layanan	Tindakan
iot	IOT: AcceptCertificateTransfer
	IOT: AddThingToBillingGroup
	IOT: AddThingToThingGroup
	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
	IOT: CreateCertificateProvider

Awalan layanan	Tindakan
	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
	IOT: CreateSecurityProfile
	IOT: CreateStream

Awalan layanan	Tindakan
	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: DeleteCustomMetric
	IOT: DeleteDimension
	IOT: DeleteDomainConfiguration
	IOT: DeleteDynamicThingGroup
	IOT: DeleteFleetMetric
	IOT: DeleteJob
	IOT: DeleteJobExecution
	IOT: DeleteJobTemplate
	IOT: DeleteMitigationAction

Awalan layanan	Tindakan
	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: DeleteV2 LoggingLevel
	IoT: DeprecateThingType
	IoT: DescribeAccountAuditConfiguration
	IoT: DescribeAuditFinding

Awalan layanan	Tindakan
	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: DescribeEndpoint
	IOT: DescribeEventConfigurations
	IOT: DescribeFleetMetric
	IOT: DescribeIndex
	IOT: DescribeJob
	IOT: DescribeJobExecution
	IOT: DescribeJobTemplate
	IOT: DescribeManagedJobTemplate

Awalan layanan	Tindakan
	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: EnableTopicRule
	IOT: GetBehaviorModelTrainingSummaries
	IOT: GetBucketsAggregation
	IOT: GetCardinality
	IOT: GetEffectivePolicies

Awalan layanan	Tindakan
	IOT: GetJobDocument
	IOT: GetLoggingOptions
	IoT: getotaupdate
	IOT: GetPackage
	IOT: GetPackageConfiguration
	IOT: GetPackageVersion
	IOT: GetPercentiles
	IOT: GetPolicy
	IOT: GetPolicyVersion
	IOT: GetRegistrationCode
	IOT: GetStatistics
	IOT: GetTopicRule
	IOT: GetTopicRuleDestination
	IoT: Getv2 LoggingOptions
	IOT: ListActiveViolations
	IOT: ListAttachedPolicies
	IOT: ListAuditFindings
	IOT: ListAuditMitigationActionsExecutions
	IOT: ListAuditMitigationActionsTasks
	IOT: ListAuditSuppressions
	IOT: ListAuditTasks

Awalan layanan	Tindakan
	IOT: ListAuthorizers
	IOT: ListBillingGroups
	IoT: ListCacertificates
	IOT: ListCertificateProviders
	IOT: ListCertificates
	IoT: CA ListCertificatesBy
	IOT: ListCustomMetrics
	IOT: ListDetectMitigationActionsExecutions
	IOT: ListDetectMitigationActionsTasks
	IOT: ListDimensions
	IOT: ListDomainConfigurations
	IOT: ListFleetMetrics
	IOT: ListIndices
	IOT: ListJobExecutionsForJob
	IOT: ListJobExecutionsForThing
	IOT: ListJobs
	IOT: ListJobTemplates
	IOT: ListManagedJobTemplates
	IOT: ListMetricValues
	IOT: ListMitigationActions
	IoT: Listotaupdates

Awalan layanan	Tindakan
	IOT: ListOutgoingCertificates
	IOT: ListPackages
	IOT: ListPackageVersions
	IOT: ListPolicies
	IOT: ListPolicyPrincipals
	IOT: ListPolicyVersions
	IOT: ListPrincipalPolicies
	IOT: ListPrincipalThings
	IOT: ListProvisioningTemplates
	IOT: ListProvisioningTemplateVersions
	IOT: ListRelatedResourcesForAuditFinding
	IOT: ListRoleAliases
	IOT: ListScheduledAudits
	IOT: ListSecurityProfiles
	IOT: ListSecurityProfilesForTarget
	IOT: ListStreams
	IOT: ListTargetsForPolicy
	IOT: ListTargetsForSecurityProfile
	IOT: ListThingGroups
	IOT: ListThingGroupsForThing
	IOT: ListThingPrincipals

Awalan layanan	Tindakan
	IOT: ListThingRegistrationTaskReports
	IOT: ListThingRegistrationTasks
	IOT: ListThings
	IOT: ListThingsInBillingGroup
	IOT: ListThingsInThingGroup
	IOT: ListThingTypes
	IOT: ListTopicRuleDestinations
	IOT: ListTopicRules
	IoT: Listv2 LoggingLevels
	IOT: ListViolationEvents
	IOT: PutVerificationStateOnViolation
	IoT: registercacertificate
	IOT: RegisterCertificate
	IoT: CA RegisterCertificateWithout
	IOT: RegisterThing
	IOT: RejectCertificateTransfer
	IOT: RemoveThingFromBillingGroup
	IOT: RemoveThingFromThingGroup
	IOT: ReplaceTopicRule
	IOT: SearchIndex
	IOT: SetDefaultAuthorizer

Awalan layanan	Tindakan
	IOT: SetDefaultPolicyVersion
	IOT: SetLoggingOptions
	IoT: Setv2 LoggingLevel
	IoT: Setv2 LoggingOptions
	IOT: StartAuditMitigationActionsTask
	IOT: StartDetectMitigationActionsTask
	IOT: StartOnDemandAuditTask
	IOT: StartThingRegistrationTask
	IOT: StopThingRegistrationTask
	IOT: TestAuthorization
	IOT: TestInvokeAuthorizer
	IOT: TransferCertificate
	IOT: UpdateAccountAuditConfiguration
	IOT: UpdateAuditSuppression
	IOT: UpdateAuthorizer
	IOT: UpdateBillingGroup
	IoT: UpdateCacertificate
	IOT: UpdateCertificate
	IOT: UpdateCertificateProvider
	IOT: UpdateCustomMetric
	IOT: UpdateDimension

Awalan layanan	Tindakan
	IOT: UpdateDomainConfiguration
	IOT: UpdateDynamicThingGroup
	IOT: UpdateEventConfigurations
	IOT: UpdateFleetMetric
	IOT: UpdateIndexingConfiguration
	IOT: UpdateJob
	IOT: UpdateMitigationAction
	IOT: UpdatePackage
	IOT: UpdatePackageConfiguration
	IOT: UpdatePackageVersion
	IOT: UpdateProvisioningTemplate
	IOT: UpdateRoleAlias
	IOT: UpdateScheduledAudit
	IOT: UpdateSecurityProfile
	IOT: UpdateStream
	IOT: UpdateThing
	IOT: UpdateThingGroup
	IOT: UpdateThingGroupsForThing
	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
	<p>iotanalitik: ListPipelines</p> <p>iotanalitik: PutLoggingOptions</p> <p>iotanalitik: RunPipelineActivity</p> <p>iotanalitik: SampleChannelData</p> <p>iotanalitik: StartPipelineReprocessing</p> <p>iotanalitik: UpdateChannel</p> <p>iotanalitik: UpdateDataset</p> <p>iotanalitik: UpdateDatastore</p> <p>iotanalitik: UpdatePipeline</p>
iotdeviceadvisor	<p>iotdeviceadvisor: CreateSuiteDefinition</p> <p>iotdeviceadvisor: DeleteSuiteDefinition</p> <p>iotdeviceadvisor: GetEndpoint</p> <p>iotdeviceadvisor: GetSuiteDefinition</p> <p>iotdeviceadvisor: GetSuiteRun</p> <p>iotdeviceadvisor: GetSuiteRunReport</p> <p>iotdeviceadvisor: ListSuiteDefinitions</p> <p>iotdeviceadvisor: ListSuiteRuns</p> <p>iotdeviceadvisor: StartSuiteRun</p> <p>iotdeviceadvisor: StopSuiteRun</p> <p>iotdeviceadvisor: UpdateSuiteDefinition</p>

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
	<p>iotevents: ListAlarmModels</p> <p>iotevents: ListAlarmModelVersions</p> <p>iotevents: ListAlarms</p> <p>iotevents: ListDetectorModels</p> <p>iotevents: ListDetectorModelVersions</p> <p>iotevents: ListDetectors</p> <p>iotevents: ListInputRoutings</p> <p>iotevents: ListInputs</p> <p>iotevents: PutLoggingOptions</p> <p>iotevents: StartDetectorModelAnalysis</p> <p>iotevents: UpdateAlarmModel</p> <p>iotevents: UpdateDetectorModel</p> <p>iotevents: UpdateInput</p>
iotfleethub	<p>iotfleethub: CreateApplication</p> <p>iotfleethub: DeleteApplication</p> <p>iotfleethub: DescribeApplication</p> <p>iotfleethub: ListApplications</p> <p>iotfleethub: UpdateApplication</p>

Awalan layanan	Tindakan
iotsitewise	iotsitewise: AssociateAssets iotsitewise: AssociateTimeSeriesToAssetProperty iotsitewise: BatchAssociateProjectAssets iotsitewise: BatchDisassociateProjectAssets iotsitewise: BatchGetAssetPropertyValue iotsitewise: BatchGetAssetPropertyValueHistory iotsitewise: BatchPutAssetPropertyValue iotsitewise: CreateAccessPolicy iotsitewise: CreateAsset iotsitewise: CreateAssetModel iotsitewise: CreateAssetModelCompositeModel iotsitewise: CreateBulkImportJob iotsitewise: CreateDashboard iotsitewise: CreateGateway iotsitewise: CreatePortal iotsitewise: CreateProject iotsitewise: DeleteAccessPolicy iotsitewise: DeleteAsset iotsitewise: DeleteAssetModel iotsitewise: DeleteAssetModelCompositeModel iotsitewise: DeleteDashboard

Awalan layanan	Tindakan
	<p>iotsitewise: DeleteGateway</p> <p>iotsitewise: DeletePortal</p> <p>iotsitewise: DeleteProject</p> <p>iotsitewise: DeleteTimeSeries</p> <p>iotsitewise: DescribeAccessPolicy</p> <p>iotsitewise: DescribeAsset</p> <p>iotsitewise: DescribeAssetCompositeModel</p> <p>iotsitewise: DescribeAssetModel</p> <p>iotsitewise: DescribeAssetModelCompositeModel</p> <p>iotsitewise: DescribeAssetProperty</p> <p>iotsitewise: DescribeBulkImportJob</p> <p>iotsitewise: DescribeDashboard</p> <p>iotsitewise: DescribeDefaultEncryptionConfiguration</p> <p>iotsitewise: DescribeGateway</p> <p>iotsitewise: DescribeGatewayCapabilityConfiguration</p> <p>iotsitewise: DescribeLoggingOptions</p> <p>iotsitewise: DescribePortal</p> <p>iotsitewise: DescribeProject</p> <p>iotsitewise: DescribeStorageConfiguration</p> <p>iotsitewise: DescribeTimeSeries</p> <p>iotsitewise: DisassociateAssets</p>

Awalan layanan	Tindakan
	<p>iotsitewise: DisassociateTimeSeriesFromAssetProperty</p> <p>iotsitewise: ExecuteAction</p> <p>iotsitewise: ExecuteQuery</p> <p>iotsitewise: ListAccessPolicies</p> <p>iotsitewise: ListActions</p> <p>iotsitewise: ListAssetModelCompositeModels</p> <p>iotsitewise: ListAssetModelProperties</p> <p>iotsitewise: ListAssetModels</p> <p>iotsitewise: ListAssetProperties</p> <p>iotsitewise: ListAssetRelationships</p> <p>iotsitewise: ListAssets</p> <p>iotsitewise: ListAssociatedAssets</p> <p>iotsitewise: ListBulkImportJobs</p> <p>iotsitewise: ListCompositionRelationships</p> <p>iotsitewise: ListDashboards</p> <p>iotsitewise: ListGateways</p> <p>iotsitewise: ListPortals</p> <p>iotsitewise: ListProjectAssets</p> <p>iotsitewise: ListProjects</p> <p>iotsitewise: ListTimeSeries</p> <p>iotsitewise: PutDefaultEncryptionConfiguration</p>

Awalan layanan	Tindakan
	<p>iotsitewise: PutLoggingOptions</p> <p>iotsitewise: PutStorageConfiguration</p> <p>iotsitewise: UpdateAccessPolicy</p> <p>iotsitewise: UpdateAsset</p> <p>iotsitewise: UpdateAssetModel</p> <p>iotsitewise: UpdateAssetModelCompositeModel</p> <p>iotsitewise: UpdateAssetProperty</p> <p>iotsitewise: UpdateDashboard</p> <p>iotsitewise: UpdateGateway</p> <p>iotsitewise: UpdateGatewayCapabilityConfiguration</p> <p>iotsitewise: UpdatePortal</p> <p>iotsitewise: UpdateProject</p>

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: ListComponents pembuat iottwinmaker: ListComponentTypes 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
	<p>iotwireless: DeleteMulticastGroup</p> <p>iotwireless: DeleteNetworkAnalyzerConfiguration</p> <p>iotwireless: DeleteQueuedMessages</p> <p>iotwireless: DeleteServiceProfile</p> <p>iotwireless: DeleteWirelessDevice</p> <p>iotwireless: DeleteWirelessDeviceImportTask</p> <p>iotwireless: DeleteWirelessGateway</p> <p>iotwireless: DeleteWirelessGatewayTask</p> <p>iotwireless: DeleteWirelessGatewayTaskDefinition</p> <p>iotwireless: DeregisterWirelessDevice</p> <p>iotwireless: DisassociateAwsAccountFromPartnerAccount</p> <p>iotwireless: DisassociateMulticastGroupFromFuotaTask</p> <p>iotwireless: DisassociateWirelessDeviceFromFuotaTask</p> <p>iotwireless: DisassociateWirelessDeviceFromMulticastGroup</p> <p>iotwireless: DisassociateWirelessDeviceFromThing</p> <p>iotwireless: DisassociateWirelessGatewayFromCertificate</p> <p>iotwireless: DisassociateWirelessGatewayFromThing</p> <p>iotwireless: GetDestination</p> <p>iotwireless: GetDeviceProfile</p> <p>iotwireless: GetEventConfigurationByResourceTypes</p> <p>iotwireless: GetFuotaTask</p>

Awalan layanan	Tindakan
	<p>iotwireless: GetLogLevelsByResourceTypes</p> <p>iotwireless: GetMetricConfiguration</p> <p>iotwireless: GetMetrics</p> <p>iotwireless: GetMulticastGroup</p> <p>iotwireless: GetMulticastGroupSession</p> <p>iotwireless: GetNetworkAnalyzerConfiguration</p> <p>iotwireless: GetPartnerAccount</p> <p>iotwireless: GetPosition</p> <p>iotwireless: GetPositionConfiguration</p> <p>iotwireless: GetPositionEstimate</p> <p>iotwireless: GetResourceEventConfiguration</p> <p>iotwireless: GetResourceLogLevel</p> <p>iotwireless: GetResourcePosition</p> <p>iotwireless: GetServiceEndpoint</p> <p>iotwireless: GetServiceProfile</p> <p>iotwireless: GetWirelessDevice</p> <p>iotwireless: GetWirelessDeviceImportTask</p> <p>iotwireless: GetWirelessDeviceStatistics</p> <p>iotwireless: GetWirelessGateway</p> <p>iotwireless: GetWirelessGatewayCertificate</p> <p>iotwireless: GetWirelessGatewayFirmwareInformation</p>

Awalan layanan	Tindakan
	<p>iotwireless: GetWirelessGatewayStatistics</p> <p>iotwireless: GetWirelessGatewayTask</p> <p>iotwireless: GetWirelessGatewayTaskDefinition</p> <p>iotwireless: ListDestinations</p> <p>iotwireless: ListDeviceProfiles</p> <p>iotwireless: ListDevicesForWirelessDeviceImportTask</p> <p>iotwireless: ListEventConfigurations</p> <p>iotwireless: ListFuotaTasks</p> <p>iotwireless: ListMulticastGroups</p> <p>iotwireless: ListMulticastGroupsByFuotaTask</p> <p>iotwireless: ListNetworkAnalyzerConfigurations</p> <p>iotwireless: ListPartnerAccounts</p> <p>iotwireless: ListPositionConfigurations</p> <p>iotwireless: ListQueuedMessages</p> <p>iotwireless: ListServiceProfiles</p> <p>iotwireless: ListWirelessDeviceImportTasks</p> <p>iotwireless: ListWirelessDevices</p> <p>iotwireless: ListWirelessGateways</p> <p>iotwireless: ListWirelessGatewayTaskDefinitions</p> <p>iotwireless: PutPositionConfiguration</p> <p>iotwireless: PutResourceLogLevel</p>

Awalan layanan	Tindakan
	<ul style="list-style-type: none"><li data-bbox="542 212 1110 247">iotwireless: ResetAllResourceLogLevels<li data-bbox="542 291 1057 327">iotwireless: ResetResourceLogLevel<li data-bbox="542 371 1101 407">iotwireless: SendDataToMulticastGroup<li data-bbox="542 451 1105 487">iotwireless: SendDataToWirelessDevice<li data-bbox="542 531 1474 567">iotwireless: StartBulkAssociateWirelessDeviceWithMulticastGroup<li data-bbox="542 611 1398 688">iotwireless: StartBulkDisassociateWirelessDeviceFromMulticastGroup<li data-bbox="542 732 927 768">iotwireless: StartFuotaTask<li data-bbox="542 812 1105 848">iotwireless: StartMulticastGroupSession<li data-bbox="542 892 1122 928">iotwireless: StartNetworkAnalyzerStream<li data-bbox="542 972 1247 1008">iotwireless: StartSingleWirelessDeviceImportTask<li data-bbox="542 1052 1159 1087">iotwireless: StartWirelessDeviceImportTask<li data-bbox="542 1131 992 1167">iotwireless: TestWirelessDevice<li data-bbox="542 1211 971 1247">iotwireless: UpdateDestination<li data-bbox="542 1291 1344 1327">iotwireless: UpdateEventConfigurationByResourceTypes<li data-bbox="542 1371 964 1407">iotwireless: UpdateFuotaTask<li data-bbox="542 1451 1219 1486">iotwireless: UpdateLogLevelsByResourceTypes<li data-bbox="542 1530 1089 1566">iotwireless: UpdateMetricConfiguration<li data-bbox="542 1610 1024 1646">iotwireless: UpdateMulticastGroup<li data-bbox="542 1690 1243 1726">iotwireless: UpdateNetworkAnalyzerConfiguration<li data-bbox="542 1770 1036 1806">iotwireless: UpdatePartnerAccount

Awalan layanan	Tindakan
	<p>iotwireless: UpdatePosition</p> <p>iotwireless: UpdateResourceEventConfiguration</p> <p>iotwireless: UpdateResourcePosition</p> <p>iotwireless: UpdateWirelessDevice</p> <p>iotwireless: UpdateWirelessDeviceImportTask</p> <p>iotwireless: UpdateWirelessGateway</p>

Awalan layanan	Tindakan
ivs	ivs: BatchGetChannel ivs: BatchGetStreamKey ivs: BatchStartViewerSessionRevocation ivs: CreateChannel ivs: CreateEncoderConfiguration ivs: CreateParticipantToken ivs: CreatePlaybackRestrictionPolicy ivs: CreateRecordingConfiguration ivs: CreateStorageConfiguration ivs: CreateStreamKey ivs: DeleteChannel ivs: DeleteEncoderConfiguration ivs: DeletePlaybackKeyPair ivs: DeletePlaybackRestrictionPolicy ivs: DeleteRecordingConfiguration ivs: DeleteStorageConfiguration ivs: DeleteStreamKey ivs: DisconnectParticipant ivs: GetChannel ivs: GetComposition ivs: GetEncoderConfiguration

Awalan layanan	Tindakan
	ivs: GetParticipant
	ivs: GetPlaybackKeyPair
	ivs: GetPlaybackRestrictionPolicy
	ivs: GetRecordingConfiguration
	ivs: GetStorageConfiguration
	ivs: GetStream
	ivs: GetStreamKey
	ivs: GetStreamSession
	ivs: ImportPlaybackKeyPair
	ivs: ListChannels
	ivs: ListCompositions
	ivs: ListEncoderConfigurations
	ivs: ListParticipantEvents
	ivs: ListParticipants
	ivs: ListPlaybackKeyPairs
	ivs: ListPlaybackRestrictionPolicies
	ivs: ListRecordingConfigurations
	ivs: ListStorageConfigurations
	ivs: ListStreamKeys
	ivs: ListStreams
	ivs: ListStreamSessions

Awalan layanan	Tindakan
	ivs: PutMetadata ivs: StartComposition ivs: StartViewerSessionRevocation ivs: StopComposition ivs: StopStream ivs: UpdateChannel 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: DescribeClusterOperation V2 kafka: DescribeCluster V2 kafka: DescribeConfiguration kafka: DescribeConfigurationRevision kafka: DescribeVpcConnection kafka: GetBootstrapBrokers kafka: GetClusterPolicy

Awalan layanan	Tindakan
	kafka: GetCompatibleKafkaVersions
	kafka: ListClientVpcConnections
	kafka: ListClusterOperations
	kafka: ListClusterOperations V2
	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: 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: 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: ListDataSources
	kendra: ListDataSourceSyncJobs
	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: 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: 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: DescribeApplicationSnapshot kinesisanalitik: DescribeApplicationVersion kinesisanalitik: DiscoverInputSchema kinesisanalitik: ListApplications

Awalan layanan	Tindakan
	kinesisanalytics: ListApplicationSnapshots
	kinesisanalytics: ListApplicationVersions
	kinesisanalytics: RollbackApplication
	kinesisanalytics: StartApplication
	kinesisanalytics: StopApplication
	kinesisanalytics: UpdateApplication
	kinesisanalytics: 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: DescribeCustomKeyStores km: DescribeKey km: DisableKey km: DisableKeyRotation km: DisconnectCustomKeyStore km: EnableKey km: EnableKeyRotation kms:Encrypt km: GenerateDataKey km: GenerateDataKeyPair km: GenerateDataKeyPairWithoutPlaintext

Awalan layanan	Tindakan
	km: GenerateDataKeyWithoutPlaintext
	km: GenerateMac
	km: GenerateRandom
	km: GetKeyPolicy
	km: GetKeyRotationStatus
	km: GetParametersForImport
	km: GetPublicKey
	km: ImportKeyMaterial
	km: ListAliases
	km: ListGrants
	km: ListKeyPolicies
	km: ListKeys
	km: ListRetirableGrants
	km: ReplicateKey
	km: RetireGrant
	km: RevokeGrant
	km: ScheduleKeyDeletion
	KMS: Tanda
	km: UpdateAlias
	km: UpdateCustomKeyStore
	km: UpdateKeyDescription

Awalan layanan	Tindakan
	km: UpdatePrimaryRegion KMS: Verifikasi km: VerifyMac

Awalan layanan	Tindakan
lambda	lambda: AddLayerVersionPermission lambda: AddLayerVersionPermission lambda: AddPermission lambda: AddPermission lambda: AddPermission lambda: CreateAlias lambda: CreateAlias lambda: CreateCodeSigningConfig lambda: CreateEventSourceMapping lambda: CreateEventSourceMapping lambda: CreateFunction lambda: CreateFunction lambda: CreateFunctionUrlConfig lambda: DeleteAlias lambda: DeleteAlias lambda: DeleteCodeSigningConfig lambda: DeleteEventSourceMapping lambda: DeleteEventSourceMapping lambda: DeleteFunction lambda: DeleteFunction lambda: DeleteFunctionCodeSigningConfig

Awalan layanan	Tindakan
	lambda: DeleteFunctionConcurrency
	lambda: DeleteFunctionConcurrency
	lambda: DeleteFunctionEventInvokeConfig
	lambda: DeleteFunctionUrlConfig
	lambda: DeleteLayerVersion
	lambda: DeleteLayerVersion
	lambda: DeleteProvisionedConcurrencyConfig
	lambda: GetAccountSettings
	lambda: GetAccountSettings
	lambda: GetAlias
	lambda: GetAlias
	lambda: GetCodeSigningConfig
	lambda: GetEventSourceMapping
	lambda: GetEventSourceMapping
	lambda: GetFunction
	lambda: GetFunction
	lambda: GetFunction
	lambda: GetFunctionCodeSigningConfig
	lambda: GetFunctionConcurrency
	lambda: GetFunctionConfiguration
	lambda: GetFunctionConfiguration

Awalan layanan	Tindakan
	lambda: GetFunctionConfiguration
	lambda: GetFunctionEventInvokeConfig
	lambda: GetFunctionUrlConfig
	lambda: GetLayerVersion
	lambda: GetLayerVersion
	lambda: GetLayerVersion
	lambda: GetLayerVersion
	lambda: GetLayerVersionPolicy
	lambda: GetLayerVersionPolicy
	lambda: GetPolicy
	lambda: GetPolicy
	lambda: GetPolicy
	lambda: GetProvisionedConcurrencyConfig
	lambda: GetRuntimeManagementConfig
	lambda: ListAliases
	lambda: ListAliases
	lambda: ListCodeSigningConfigs
	lambda: ListEventSourceMappings
	lambda: ListEventSourceMappings
	lambda: ListFunctionEventInvokeConfigs
	lambda: ListFunctions

Awalan layanan	Tindakan
	lambda: ListFunctions
	lambda: ListFunctionsByCodeSigningConfig
	lambda: ListFunctionUrlConfigs
	lambda: ListLayers
	lambda: ListLayers
	lambda: ListLayerVersions
	lambda: ListLayerVersions
	lambda: ListProvisionedConcurrencyConfigs
	lambda: ListVersionsByFunction
	lambda: ListVersionsByFunction
	lambda: PublishLayerVersion
	lambda: PublishLayerVersion
	lambda: PublishVersion
	lambda: PublishVersion
	lambda: PutFunctionCodeSigningConfig
	lambda: PutFunctionConcurrency
	lambda: PutFunctionConcurrency
	lambda: PutFunctionEventInvokeConfig
	lambda: PutProvisionedConcurrencyConfig
	lambda: PutRuntimeManagementConfig
	lambda: RemoveLayerVersionPermission

Awalan layanan	Tindakan
	lambda: RemoveLayerVersionPermission
	lambda: RemovePermission
	lambda: RemovePermission
	lambda: RemovePermission
	lambda: UpdateAlias
	lambda: UpdateAlias
	lambda: UpdateCodeSigningConfig
	lambda: UpdateEventSourceMapping
	lambda: UpdateEventSourceMapping
	lambda: UpdateFunctionCode
	lambda: UpdateFunctionCode
	lambda: UpdateFunctionCode
	lambda: UpdateFunctionConfiguration
	lambda: UpdateFunctionConfiguration
	lambda: UpdateFunctionConfiguration
	lambda: UpdateFunctionEventInvokeConfig
	lambda: UpdateFunctionUrlConfig

Awalan layanan	Tindakan
lex	lex: BatchCreateCustomVocabularyItem lex: BatchDeleteCustomVocabularyItem lex: BatchUpdateCustomVocabularyItem lex: BuildBotLocale lex: CreateBotAlias lex: CreateBotVersion lex: CreateExport lex: CreateIntentVersion lex: CreateResourcePolicy lex: CreateSlotTypeVersion lex: CreateTestSetDiscrepancyReport lex: CreateUploadUrl lex: DeleteBot lex: DeleteBotChannelAssociation lex: DeleteExport lex: DeleteImport lex: DeleteIntentVersion lex: DeleteResourcePolicy lex: DeleteSlotTypeVersion lex: DeleteTestSet lex: DeleteUtterances

Awalan layanan	Tindakan
	<p>lex: DescribeBotAlias</p> <p>lex: DescribeBotRecommendation</p> <p>lex: DescribeBotResourceGeneration</p> <p>lex: DescribeBotVersion</p> <p>lex: DescribeCustomVocabularyMetadata</p> <p>lex: DescribeExport</p> <p>lex: DescribeImport</p> <p>lex: DescribeResourcePolicy</p> <p>lex: DescribeTestExecution</p> <p>lex: DescribeTestSet</p> <p>lex: DescribeTestSetDiscrepancyReport</p> <p>lex: DescribeTestSetGeneration</p> <p>lex: GenerateBotElement</p> <p>lex: GetBot</p> <p>lex: GetBotAlias</p> <p>lex: GetBotAliases</p> <p>lex: GetBotChannelAssociation</p> <p>lex: GetBotChannelAssociations</p> <p>lex: GetBots</p> <p>lex: GetBotVersions</p> <p>lex: GetBuiltinIntent</p>

Awalan layanan	Tindakan
	lex: GetBuiltinIntents
	lex: GetBuiltinSlotTypes
	lex: GetExport
	lex: GetImport
	lex: GetIntent
	lex: GetIntents
	lex: GetIntentVersions
	lex: GetMigration
	lex: GetMigrations
	lex: GetSlotType
	lex: GetSlotTypes
	lex: GetSlotTypeVersions
	lex: GetTestExecutionArtifactsUrl
	lex: GetUtterancesView
	lex: ListBotAliases
	lex: ListBotRecommendations
	lex: ListBotResourceGenerations
	lex: ListBots
	lex: ListBotVersions
	lex: ListBuiltinIntents
	lex: ListBuiltinSlotTypes

Awalan layanan	Tindakan
	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
	lex: PutIntent
	lex: PutSlotType
	lex: SearchAssociatedTranscripts
	lex: StartBotRecommendation
	lex: StartImport
	lex: StartMigration
	lex: StartTestExecution
	lex: StartTestSetGeneration

Awalan layanan	Tindakan
	lex: StopBotRecommendation lex: UpdateBotAlias lex: UpdateBotRecommendation lex: UpdateExport lex: UpdateResourcePolicy
license-manager-linux-subscriptions	license-manager-linux-subscriptions:GetServiceSettings license-manager-linux-subscriptions:ListLinuxSubscriptionInstances license-manager-linux-subscriptions:ListLinuxSubscriptions license-manager-linux-subscriptions: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
	<p>LightSail: createGUI SessionAccessDetails</p> <p>lightsail: CreateInstances</p> <p>lightsail: CreateInstancesFromSnapshot</p> <p>lightsail: CreateInstanceSnapshot</p> <p>lightsail: CreateKeyPair</p> <p>lightsail: CreateLoadBalancer</p> <p>lightsail: CreateLoadBalancerTlsCertificate</p> <p>lightsail: CreateRelationalDatabase</p> <p>lightsail: CreateRelationalDatabaseFromSnapshot</p> <p>lightsail: CreateRelationalDatabaseSnapshot</p> <p>lightsail: DeleteAlarm</p> <p>lightsail: DeleteAutoSnapshot</p> <p>lightsail: DeleteBucket</p> <p>lightsail: DeleteBucketAccessKey</p> <p>lightsail: DeleteCertificate</p> <p>lightsail: DeleteContactMethod</p> <p>lightsail: DeleteContainerImage</p> <p>lightsail: DeleteContainerService</p> <p>lightsail: DeleteDisk</p> <p>lightsail: DeleteDiskSnapshot</p> <p>lightsail: DeleteDistribution</p>

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: ApiMetadata GetContainer
	lightsail: GetContainerImages
	lightsail: GetContainerLog
	lightsail: GetContainerServiceDeployments
	lightsail: GetContainerServiceMetricData
	lightsail: GetContainerServicePowers
	lightsail: GetContainerServices
	lightsail: GetCostEstimate
	lightsail: GetDisk
	lightsail: GetDisks
	lightsail: GetDiskSnapshot
	lightsail: GetDiskSnapshots

Awalan layanan	Tindakan
	lightsail: GetDistributionBundles
	lightsail: GetDistributionLatestCacheReset
	lightsail: GetDistributionMetricData
	lightsail: GetDistributions
	lightsail: GetDomain
	lightsail: GetExportSnapshotRecords
	lightsail: GetInstance
	lightsail: GetInstanceMetricData
	lightsail: GetInstancePortStates
	lightsail: GetInstances
	lightsail: GetInstanceSnapshot
	lightsail: GetInstanceSnapshots
	lightsail: GetInstanceState
	lightsail: GetKeyPair
	lightsail: GetKeyPairs
	lightsail: GetLoadBalancer
	lightsail: GetLoadBalancerMetricData
	lightsail: GetLoadBalancers
	lightsail: GetLoadBalancerTlsCertificates
	lightsail: GetLoadBalancerTlsPolicies
	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: GetRelationalDatabases
	lightsail: GetRelationalDatabaseSnapshot
	lightsail: GetRelationalDatabaseSnapshots
	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: CancelExportTask log: CreateExportTask log: CreateLogAnomalyDetector log: CreateLogGroup log: CreateLogStream log: DeleteDataProtectionPolicy log: DeleteDelivery log: DeleteDeliveryDestination log: DeleteDeliveryDestinationPolicy log: DeleteDeliverySource log: DeleteDestination log: DeleteLogGroup log: DeleteLogStream log: DeleteMetricFilter log: DeleteQueryDefinition log: DeleteResourcePolicy log: DeleteRetentionPolicy log: DeleteSubscriptionFilter log: DescribeAccountPolicies log: DescribeDeliveries

Awalan layanan	Tindakan
	log: DescribeDeliveryDestinations
	log: DescribeDeliverySources
	log: DescribeDestinations
	log: DescribeExportTasks
	log: DescribeLogGroups
	log: DescribeLogStreams
	log: DescribeMetricFilters
	log: DescribeQueries
	log: DescribeQueryDefinitions
	log: DescribeResourcePolicies
	log: DescribeSubscriptionFilters
	log: DisassociateKmsKey
	log: GetDataProtectionPolicy
	log: GetDelivery
	log: GetDeliveryDestination
	log: GetDeliveryDestinationPolicy
	log: GetDeliverySource
	log: GetLogGroupFields
	log: GetLogRecord
	log: GetQueryResults
	log: ListAnomalies

Awalan layanan	Tindakan
	log: ListLogAnomalyDetectors
	log: PutDataProtectionPolicy
	log: PutDeliveryDestination
	log: PutDeliveryDestinationPolicy
	log: PutDeliverySource
	log: PutDestination
	log: PutDestinationPolicy
	log: PutMetricFilter
	log: PutQueryDefinition
	log: PutResourcePolicy
	log: PutRetentionPolicy
	log: PutSubscriptionFilter
	log: StartLiveTail
	log: StartQuery
	log: StopQuery
	log: TestMetricFilter

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: DescribeLabel Lookoutequipment: DescribeLabelGroup Lookoutequipment: DescribeModel Lookoutequipment: DescribeModelVersion Lookoutequipment: DescribeResourcePolicy Lookoutequipment: DescribeRetrainingScheduler

Awalan layanan	Tindakan
	Lookoutequipment: ImportDataset
	Lookoutequipment: ImportModelVersion
	Lookoutequipment: ListDataIngestionJobs
	Lookoutequipment: ListDatasets
	peralatan pencarian: ListInferenceEvents
	peralatan pencarian: ListInferenceExecutions
	peralatan pencarian: ListInferenceSchedulers
	peralatan pencarian: ListLabelGroups
	peralatan pencarian: ListLabels
	peralatan pencarian: ListModels
	peralatan pencarian: ListModelVersions
	peralatan pencarian: ListRetrainingSchedulers
	peralatan pencarian: ListSensorStatistics
	peralatan pencarian: PutResourcePolicy
	peralatan pencarian: StartDataIngestionJob
	peralatan pencarian: StartInferenceScheduler
	peralatan pencarian: StartRetrainingScheduler
	peralatan pencarian: StopInferenceScheduler
	peralatan pencarian: StopRetrainingScheduler
	peralatan pencarian: UpdateActiveModelVersion
	peralatan pencarian: UpdateInferenceScheduler

Awalan layanan	Tindakan
	peralatan pencarian: UpdateLabelGroup peralatan pencarian: UpdateModel peralatan pencarian: 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
	<p>lookoutmetrics: ListAnomalyGroupTimeSeries</p> <p>lookoutmetrics: ListMetricSets</p> <p>lookoutmetrics: PutFeedback</p> <p>lookoutmetrics: UpdateAlert</p> <p>lookoutmetrics: UpdateAnomalyDetector</p> <p>lookoutmetrics: UpdateMetricSet</p>

Awalan layanan	Tindakan
lookoutvision	penglihatan penglihatan: CreateDataset penglihatan penglihatan: CreateModel penglihatan penglihatan: CreateProject penglihatan penglihatan: DeleteDataset penglihatan penglihatan: DeleteModel penglihatan penglihatan: DeleteProject penglihatan penglihatan: DescribeDataset penglihatan penglihatan: DescribeModel penglihatan penglihatan: DescribeModelPackagingJob penglihatan penglihatan: DescribeProject penglihatan penglihatan: DetectAnomalies penglihatan penglihatan: ListDatasetEntries penglihatan penglihatan: ListModelPackagingJobs penglihatan penglihatan: ListModels penglihatan penglihatan: ListProjects penglihatan penglihatan: StartModel penglihatan penglihatan: StartModelPackagingJob penglihatan penglihatan: StopModel penglihatan penglihatan: UpdateDatasetEntries

Awalan layanan	Tindakan
m2	m2: CancelBatchJobExecution m2: CreateApplication m2: CreateDataSetImportTask m2: CreateDeployment m2: CreateEnvironment m2: DeleteApplication m2: DeleteApplicationFromEnvironment m2: DeleteEnvironment m2: GetApplication m2: GetApplicationVersion m2: GetBatchJobExecution m2: GetDataSetDetails m2: GetDataSetImportTask m2: GetDeployment m2: GetEnvironment m2: GetSignedBluinsightsUrl m2: ListApplications m2: ListApplicationVersions m2: ListBatchJobDefinitions m2: ListBatchJobExecutions m2: ListDataSetImportHistory

Awalan layanan	Tindakan
	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: ListProposals

Awalan layanan	Tindakan
	terkelolablockchain: ListProposalVotes terkelolablockchain: RejectInvitation terkelolablockchain: UpdateMember terkelolablockchain: UpdateNode terkelolablockchain: VoteOnProposal

Awalan layanan	Tindakan
mediacconnect	mediacconnect: AddBridgeOutputs mediacconnect: AddBridgeSources mediacconnect: AddFlowMediaStreams mediacconnect: AddFlowOutputs mediacconnect: AddFlowSources mediacconnect: AddFlowVpclInterfaces mediacconnect: CreateBridge mediacconnect: CreateFlow mediacconnect: CreateGateway mediacconnect: DeleteBridge mediacconnect: DeleteFlow mediacconnect: DeleteGateway mediacconnect: DeregisterGatewayInstance mediacconnect: DescribeBridge mediacconnect: DescribeFlow mediacconnect: DescribeFlowSourceMetadata mediacconnect: DescribeGateway mediacconnect: DescribeGatewayInstance mediacconnect: DescribeOffering mediacconnect: DescribeReservation mediacconnect: GrantFlowEntitlements

Awalan layanan	Tindakan
	<code>mediacconnect: ListBridges</code>
	<code>mediacconnect: ListEntitlements</code>
	<code>mediacconnect: ListFlows</code>
	<code>mediacconnect: ListGatewayInstances</code>
	<code>mediacconnect: ListGateways</code>
	<code>mediacconnect: ListOfferings</code>
	<code>mediacconnect: ListReservations</code>
	<code>mediacconnect: PurchaseOffering</code>
	<code>mediacconnect: RemoveBridgeOutput</code>
	<code>mediacconnect: RemoveBridgeSource</code>
	<code>mediacconnect: RemoveFlowMediaStream</code>
	<code>mediacconnect: RemoveFlowOutput</code>
	<code>mediacconnect: RemoveFlowSource</code>
	<code>mediacconnect: RemoveFlowVpclInterface</code>
	<code>mediacconnect: RevokeFlowEntitlement</code>
	<code>mediacconnect: StartFlow</code>
	<code>mediacconnect: StopFlow</code>
	<code>mediacconnect: UpdateBridge</code>
	<code>mediacconnect: UpdateBridgeOutput</code>
	<code>mediacconnect: UpdateBridgeSource</code>
	<code>mediacconnect: UpdateBridgeState</code>

Awalan layanan	Tindakan
	<p>mediacconnect: UpdateFlow</p> <p>mediacconnect: UpdateFlowEntitlement</p> <p>mediacconnect: UpdateFlowMediaStream</p> <p>mediacconnect: UpdateFlowOutput</p> <p>mediacconnect: UpdateFlowSource</p> <p>mediacconnect: UpdateGatewayInstance</p>

Awalan layanan	Tindakan
mediaconvert	mediaconvert: AssociateCertificate mediaconvert: CancelJob mediaconvert: CreateJob mediaconvert: CreateJobTemplate mediaconvert: CreatePreset mediaconvert: CreateQueue mediaconvert: DeleteJobTemplate mediaconvert: DeletePolicy mediaconvert: DeletePreset mediaconvert: DeleteQueue mediaconvert: DescribeEndpoints mediaconvert: DisassociateCertificate mediaconvert: GetJob mediaconvert: GetJobTemplate mediaconvert: GetPolicy mediaconvert: GetPreset mediaconvert: GetQueue mediaconvert: ListJobs mediaconvert: ListJobTemplates mediaconvert: ListPresets mediaconvert: ListQueues

Awalan layanan	Tindakan
	mediaconvert: PutPolicy mediaconvert: UpdateJobTemplate mediaconvert: UpdatePreset mediaconvert: UpdateQueue

Awalan layanan	Tindakan
medialive	medialive: AcceptInputDeviceTransfer medialive: BatchDelete medialive: BatchStart medialive: BatchStop medialive: BatchUpdateSchedule medialive: CancellInputDeviceTransfer medialive: ClaimDevice medialive: CreateChannel medialive: CreateCloudWatchAlarmTemplate medialive: CreateCloudWatchAlarmTemplateGroup medialive: CreateEventBridgeRuleTemplate medialive: CreateEventBridgeRuleTemplateGroup medialive: CreateInput medialive: CreateInputSecurityGroup medialive: CreateMultiplex medialive: CreateMultiplexProgram medialive: CreatePartnerInput medialive: CreateSignalMap medialive: DeleteChannel medialive: DeleteCloudWatchAlarmTemplate medialive: DeleteCloudWatchAlarmTemplateGroup

Awalan layanan	Tindakan
	medialive: DeleteEventBridgeRuleTemplate
	medialive: DeleteEventBridgeRuleTemplateGroup
	medialive: DeleteInput
	medialive: DeleteInputSecurityGroup
	medialive: DeleteMultiplex
	medialive: DeleteMultiplexProgram
	medialive: DeleteReservation
	medialive: DeleteSchedule
	medialive: DeleteSignalMap
	medialive: DescribeAccountConfiguration
	medialive: DescribeChannel
	medialive: DescribeInput
	medialive: DescribeInputDevice
	medialive: DescribeInputDeviceThumbnail
	medialive: DescribeInputSecurityGroup
	medialive: DescribeMultiplex
	medialive: DescribeMultiplexProgram
	medialive: DescribeOffering
	medialive: DescribeReservation
	medialive: DescribeSchedule
	medialive: DescribeThumbnails

Awalan layanan	Tindakan
	medialive: GetCloudWatchAlarmTemplate
	medialive: GetCloudWatchAlarmTemplateGroup
	medialive: GetEventBridgeRuleTemplate
	medialive: GetEventBridgeRuleTemplateGroup
	medialive: GetSignalMap
	medialive: ListChannels
	medialive: ListCloudWatchAlarmTemplateGroups
	medialive: ListCloudWatchAlarmTemplates
	medialive: ListEventBridgeRuleTemplateGroups
	medialive: ListEventBridgeRuleTemplates
	medialive: ListInputDevices
	medialive: ListInputDeviceTransfers
	medialive: ListInputs
	medialive: ListInputSecurityGroups
	medialive: ListMultiplexes
	medialive: ListMultiplexPrograms
	medialive: ListOfferings
	medialive: ListReservations
	medialive: ListSignalMaps
	medialive: PurchaseOffering
	medialive: RebootInputDevice

Awalan layanan	Tindakan
	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: UpdateCloudWatchAlarmTemplate
	medialive: UpdateCloudWatchAlarmTemplateGroup
	medialive: UpdateEventBridgeRuleTemplate
	medialive: UpdateEventBridgeRuleTemplateGroup
	medialive: UpdateInput

Awalan layanan	Tindakan
	<p>medialive: UpdateInputDevice</p> <p>medialive: UpdateInputSecurityGroup</p> <p>medialive: UpdateMultiplex</p> <p>medialive: UpdateMultiplexProgram</p> <p>medialive: UpdateReservation</p>
mediapackage	<p>mediapackage: ConfigureLogs</p> <p>mediapackage: CreateChannel</p> <p>mediapackage: CreateHarvestJob</p> <p>mediapackage: CreateOriginEndpoint</p> <p>mediapackage: DeleteChannel</p> <p>mediapackage: DeleteOriginEndpoint</p> <p>mediapackage: DescribeChannel</p> <p>mediapackage: DescribeHarvestJob</p> <p>mediapackage: DescribeOriginEndpoint</p> <p>mediapackage: ListChannels</p> <p>mediapackage: ListHarvestJobs</p> <p>mediapackage: ListOriginEndpoints</p> <p>mediapackage: RotateChannelCredentials</p> <p>mediapackage: RotateIngestEndpointCredentials</p> <p>mediapackage: UpdateChannel</p> <p>mediapackage: UpdateOriginEndpoint</p>

Awalan layanan	Tindakan
mediapackage-vod	mediapackage-vod: ConfigureLogs mediapackage-vod: CreateAsset mediapackage-vod: CreatePackagingConfiguration mediapackage-vod: CreatePackagingGroup mediapackage-vod: DeleteAsset mediapackage-vod: DeletePackagingConfiguration mediapackage-vod: DeletePackagingGroup mediapackage-vod: DescribeAsset mediapackage-vod: DescribePackagingConfiguration mediapackage-vod: DescribePackagingGroup mediapackage-vod: ListAssets mediapackage-vod: ListPackagingConfigurations mediapackage-vod: ListPackagingGroups mediapackage-vod: UpdatePackagingGroup

Awalan layanan	Tindakan
mediastore	mediastore: CreateContainer mediastore: DeleteContainer mediastore: DeleteContainerPolicy mediastore: DeleteCorsPolicy mediastore: DeleteLifecyclePolicy mediastore: DeleteMetricPolicy mediastore: DescribeContainer mediastore: GetContainerPolicy mediastore: GetCorsPolicy mediastore: GetLifecyclePolicy mediastore: GetMetricPolicy mediastore: ListContainers mediastore: PutContainerPolicy mediastore: PutCorsPolicy mediastore: PutLifecyclePolicy mediastore: PutMetricPolicy mediastore: StartAccessLogging mediastore: StopAccessLogging

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: CreateParameterGroup memorydb: CreateSnapshot memorydb: CreateSubnetGroup memorydb: CreateUser memorydb: DeleteAcl memorydb: DeleteCluster memorydb: DeleteParameterGroup memorydb: DeleteSnapshot memorydb: DeleteSubnetGroup memorydb: DeleteUser memorydb: DescribeAcls memorydb: DescribeClusters memorydb: DescribeEngineVersions memorydb: DescribeEvents memorydb: DescribeParameterGroups memorydb: DescribeParameters memorydb: DescribeReservedNodes

Awalan layanan	Tindakan
	memorydb: DescribeReservedNodesOfferings
	memorydb: DescribeServiceUpdates
	memorydb: DescribeSnapshots
	memorydb: DescribeSubnetGroups
	memorydb: DescribeUsers
	memorydb: FailoverShard
	memorydb: ListAllowedNodeTypeUpdates
	memorydb: PurchaseReservedNodesOffering
	memorydb: ResetParameterGroup
	memorydb: UpdateAcl
	memorydb: UpdateCluster
	memorydb: UpdateParameterGroup
	memorydb: UpdateSubnetGroup
	memorydb: UpdateUser

Awalan layanan	Tindakan
mgh	mgh: AssociateCreatedArtifact mgh: AssociateDiscoveredResource mgh: CreateHomeRegionControl mgh: CreateProgressUpdateStream mgh: DeleteHomeRegionControl mgh: DeleteProgressUpdateStream mgh: DescribeApplicationState mgh: DescribeHomeRegionControls mgh: DescribeMigrationTask mgh: DisassociateCreatedArtifact mgh: DisassociateDiscoveredResource mgh: GetHomeRegion mgh: ImportMigrationTask mgh: ListApplicationStates mgh: ListCreatedArtifacts mgh: ListDiscoveredResources mgh: ListMigrationTasks mgh: ListProgressUpdateStreams 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: CreateReplicationConfigurationTemplate mgn: CreateWave mgn: DeleteApplication mgn: DeleteConnector mgn: DeleteJob mgn: DeleteLaunchConfigurationTemplate mgn: DeleteReplicationConfigurationTemplate mgn: DeleteSourceServer mgn: DeleteVcenterClient mgn: DeleteWave mgn: DescribeJobLogItems mgn: DescribeJobs mgn: DescribeLaunchConfigurationTemplates

Awalan layanan	Tindakan
	<ul style="list-style-type: none">mgn: DescribeReplicationConfigurationTemplatesmgn: DescribeVcenterClientsmgn: DisassociateApplicationsmgn: DisassociateSourceServersmgn: DisconnectFromServicemgn: FinalizeCutovermgn: GetReplicationConfigurationmgn: InitializeServicemgn: ListConnectorsmgn: ListExportErrorsmgn: ListExportsmgn: ListImportErrorsmgn: ListImportsmgn: ListManagedAccountsmgn: ListSourceServerActionsmgn: ListTemplateActionsmgn: MarkAsArchivedmgn: PauseReplicationmgn: PutSourceServerActionmgn: PutTemplateActionmgn: RemoveSourceServerAction

Awalan layanan	Tindakan
	<p>mgn: RemoveTemplateAction</p> <p>mgn: ResumeReplication</p> <p>mgn: RetryDataReplication</p> <p>mgn: StartCutover</p> <p>mgn: StartExport</p> <p>mgn: StartImport</p> <p>mgn: StartReplication</p> <p>mgn: StartTest</p> <p>mgn: StopReplication</p> <p>mgn: TerminateTargetInstances</p> <p>mgn: UnarchiveApplication</p> <p>mgn: UnarchiveWave</p> <p>mgn: UpdateApplication</p> <p>mgn: UpdateConnector</p> <p>mgn: UpdateLaunchConfigurationTemplate</p> <p>mgn: UpdateReplicationConfiguration</p> <p>mgn: UpdateReplicationConfigurationTemplate</p> <p>mgn: UpdateSourceServer</p> <p>mgn: UpdateSourceServerReplicationType</p> <p>mgn: UpdateWave</p>

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: PutPortfolioPreferences strategi hub migrasi: RegisterCollector

Awalan layanan	Tindakan
	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
	<p>penargetan seluler: <code>GetApplicationSettings</code></p> <p>penargetan seluler: <code>GetApps</code></p> <p>penargetan seluler: <code>GetBaiduChannel</code></p> <p>penargetan seluler: <code>GetCampaign</code></p> <p>penargetan seluler: <code>GetCampaignActivities</code></p> <p>penargetan seluler: <code>GetCampaignDateRangeKpi</code></p> <p>penargetan seluler: <code>GetCampaigns</code></p> <p>penargetan seluler: <code>GetCampaignVersion</code></p> <p>penargetan seluler: <code>GetCampaignVersions</code></p> <p>penargetan seluler: <code>GetChannels</code></p> <p>penargetan seluler: <code>GetEmailChannel</code></p> <p>penargetan seluler: <code>GetEmailTemplate</code></p> <p>penargetan seluler: <code>GetEndpoint</code></p> <p>penargetan seluler: <code>GetEventStream</code></p> <p>penargetan seluler: <code>GetExportJob</code></p> <p>penargetan seluler: <code>GetExportJobs</code></p> <p>penargetan seluler: <code>GetGcmChannel</code></p> <p>penargetan seluler: <code>GetImportJob</code></p> <p>penargetan seluler: <code>GetImportJobs</code></p> <p>penargetan seluler: <code>GetInAppMessages</code></p> <p>penargetan seluler: <code>GetInAppTemplate</code></p>

Awalan layanan	Tindakan
	<p>penargetan seluler: GetJourney</p> <p>penargetan seluler: GetJourneyDateRangeKpi</p> <p>penargetan seluler: GetJourneyExecutionActivityMetrics</p> <p>penargetan seluler: GetJourneyExecutionMetrics</p> <p>penargetan seluler: GetJourneyRunExecutionActivityMetrics</p> <p>penargetan seluler: GetJourneyRunExecutionMetrics</p> <p>penargetan seluler: GetJourneyRuns</p> <p>penargetan seluler: GetPushTemplate</p> <p>penargetan seluler: GetRecommenderConfiguration</p> <p>penargetan seluler: GetRecommenderConfigurations</p> <p>penargetan seluler: GetSegment</p> <p>penargetan seluler: GetSegmentExportJobs</p> <p>penargetan seluler: GetSegmentImportJobs</p> <p>penargetan seluler: GetSegments</p> <p>penargetan seluler: GetSegmentVersion</p> <p>penargetan seluler: GetSegmentVersions</p> <p>penargetan seluler: GetSmsChannel</p> <p>penargetan seluler: GetSmsTemplate</p> <p>penargetan seluler: GetUserEndpoints</p> <p>penargetan seluler: GetVoiceChannel</p> <p>penargetan seluler: GetVoiceTemplate</p>

Awalan layanan	Tindakan
	<p>penargetan seluler: ListJourneys</p> <p>penargetan seluler: ListTemplates</p> <p>penargetan seluler: ListTemplateVersions</p> <p>penargetan seluler: PhoneNumberValidate</p> <p>penargetan seluler: PutEventStream</p> <p>penargetan seluler: RemoveAttributes</p> <p>penargetan seluler: UpdateAdmChannel</p> <p>penargetan seluler: UpdateApnsChannel</p> <p>penargetan seluler: UpdateApnsSandboxChannel</p> <p>penargetan seluler: UpdateApnsVoipChannel</p> <p>penargetan seluler: UpdateApnsVoipSandboxChannel</p> <p>penargetan seluler: UpdateApplicationSettings</p> <p>penargetan seluler: UpdateBaiduChannel</p> <p>penargetan seluler: UpdateCampaign</p> <p>penargetan seluler: UpdateEmailChannel</p> <p>penargetan seluler: UpdateEmailTemplate</p> <p>penargetan seluler: UpdateEndpoint</p> <p>penargetan seluler: UpdateEndpointsBatch</p> <p>penargetan seluler: UpdateGcmChannel</p> <p>penargetan seluler: UpdateInAppTemplate</p> <p>penargetan seluler: UpdateJourney</p>

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: DeleteUser mq: DescribeBroker mq: DescribeBrokerEngineTypes mq: DescribeBrokerInstanceOptions mq: DescribeConfiguration mq: DescribeConfigurationRevision mq: DescribeUser mq: ListBrokers mq: ListConfigurationRevisions mq: ListConfigurations mq: ListUsers MQ: Mempromosikan mq: RebootBroker mq: UpdateBroker mq: UpdateConfiguration 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: CreateConnection manajer jaringan: CreateConnectPeer manajer jaringan: CreateCoreNetwork manajer jaringan: CreateDevice 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: DeleteConnection manajer jaringan: DeleteConnectPeer manajer jaringan: DeleteCoreNetwork

Awalan layanan	Tindakan
	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: GetConnections
	manajer jaringan: GetConnectPeer
	manajer jaringan: GetConnectPeerAssociations
	manajer jaringan: GetCoreNetwork
	manajer jaringan: GetCoreNetworkChangeEvents
	manajer jaringan: GetCoreNetworkChangeSet

Awalan layanan	Tindakan
	<p>manajer jaringan: GetCoreNetworkPolicy</p> <p>manajer jaringan: GetCustomerGatewayAssociations</p> <p>manajer jaringan: GetDevices</p> <p>manajer jaringan: GetLinkAssociations</p> <p>manajer jaringan: GetLinks</p> <p>manajer jaringan: GetNetworkResourceCounts</p> <p>manajer jaringan: GetNetworkResourceRelationships</p> <p>manajer jaringan: GetNetworkResources</p> <p>manajer jaringan: GetNetworkRoutes</p> <p>manajer jaringan: GetNetworkTelemetry</p> <p>manajer jaringan: GetResourcePolicy</p> <p>manajer jaringan: GetRouteAnalysis</p> <p>manajer jaringan: GetSites</p> <p>manajer jaringan: GetSiteToSiteVpnAttachment</p> <p>manajer jaringan: GetTransitGatewayConnectPeerAssociations</p> <p>manajer jaringan: GetTransitGatewayPeering</p> <p>manajer jaringan: GetTransitGatewayRegistrations</p> <p>manajer jaringan: GetTransitGatewayRouteTableAttachment</p> <p>manajer jaringan: GetVpcAttachment</p> <p>manajer jaringan: ListAttachments</p> <p>manajer jaringan: ListConnectPeers</p>

Awalan layanan	Tindakan
	<p>manajer jaringan: ListCoreNetworkPolicyVersions</p> <p>manajer jaringan: ListCoreNetworks</p> <p>manajer jaringan: ListOrganizationServiceAccessStatus</p> <p>manajer jaringan: ListPeerings</p> <p>manajer jaringan: PutCoreNetworkPolicy</p> <p>manajer jaringan: PutResourcePolicy</p> <p>manajer jaringan: RegisterTransitGateway</p> <p>manajer jaringan: RejectAttachment</p> <p>manajer jaringan: RestoreCoreNetworkPolicyVersion</p> <p>manajer jaringan: StartOrganizationServiceAccessUpdate</p> <p>manajer jaringan: StartRouteAnalysis</p> <p>manajer jaringan: UpdateConnection</p> <p>manajer jaringan: UpdateCoreNetwork</p> <p>manajer jaringan: UpdateDevice</p> <p>manajer jaringan: UpdateGlobalNetwork</p> <p>manajer jaringan: UpdateLink</p> <p>manajer jaringan: UpdateNetworkResourceMetadata</p> <p>manajer jaringan: UpdateSite</p> <p>manajer jaringan: UpdateVpcAttachment</p>

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: SSO StartStudio ConfigurationRepair
	gesit: StopStreamingSession
	gesit: UpdateLaunchProfile
	gesit: UpdateLaunchProfileMember
	gesit: UpdateStreamingImage
	gesit: UpdateStudio
	gesit: UpdateStudioComponent

Awalan layanan	Tindakan
omics	omics: AbortMultipartReadSetUpload omics: BatchDeleteReadSet omics: CancelAnnotationImportJob omics: CancelRun omics: CancelVariantImportJob omics: CompleteMultipartReadSetUpload omics: CreateAnnotationStore omics: CreateMultipartReadSetUpload omics: CreateReferenceStore omics: CreateRunGroup omics: CreateSequenceStore omics: CreateVariantStore omics: CreateWorkflow omics: DeleteAnnotationStore omics: DeleteReference omics: DeleteReferenceStore omics: DeleteRun omics: DeleteRunGroup omics: DeleteSequenceStore omics: DeleteVariantStore omics: DeleteWorkflow

Awalan layanan	Tindakan
	<p>omics: GetAnnotationImportJob</p> <p>omics: GetAnnotationStore</p> <p>omics: GetReadSet</p> <p>omics: GetReadSetActivationJob</p> <p>omics: GetReadSetExportJob</p> <p>omics: GetReadSetImportJob</p> <p>omics: GetReadSetMetadata</p> <p>omics: GetReference</p> <p>omics: GetReferenceImportJob</p> <p>omics: GetReferenceMetadata</p> <p>omics: GetReferenceStore</p> <p>omics: GetRun</p> <p>omics: GetRunGroup</p> <p>omics: GetRunTask</p> <p>omics: GetSequenceStore</p> <p>omics: GetVariantImportJob</p> <p>omics: GetVariantStore</p> <p>omics: GetWorkflow</p> <p>omics: ListAnnotationImportJobs</p> <p>omics: ListAnnotationStores</p> <p>omics: ListMultipartReadSetUploads</p>

Awalan layanan	Tindakan
	<p>omics: ListReadSetActivationJobs</p> <p>omics: ListReadSetExportJobs</p> <p>omics: ListReadSetImportJobs</p> <p>omics: ListReadSets</p> <p>omics: ListReadSetUploadParts</p> <p>omics: ListReferenceImportJobs</p> <p>omics: ListReferences</p> <p>omics: ListReferenceStores</p> <p>omics: ListRunGroups</p> <p>omics: ListRuns</p> <p>omics: ListRunTasks</p> <p>omics: ListSequenceStores</p> <p>omics: ListVariantImportJobs</p> <p>omics: ListVariantStores</p> <p>omics: ListWorkflows</p> <p>omics: StartAnnotationImportJob</p> <p>omics: StartReadSetActivationJob</p> <p>omics: StartReadSetExportJob</p> <p>omics: StartReadSetImportJob</p> <p>omics: StartReferenceImportJob</p> <p>omics: StartRun</p>

Awalan layanan	Tindakan
	omics: StartVariantImportJob omics: UpdateAnnotationStore omics: UpdateRunGroup omics: UpdateVariantStore omics: UpdateWorkflow 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: DescribeStacks
	opsworks: DescribeStackSummary
	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
	<ul style="list-style-type: none">opsworks: UpdateElasticIpopsworks: UpdateInstanceopsworks: UpdateLayeropsworks: UpdateMyUserProfileopsworks: UpdateRdsDbInstanceopsworks: UpdateStackopsworks: UpdateUserProfileopsworks: 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: DetachPolicy
	Organisasi: Nonaktifkan AWSServiceAccess
	organisasi: DisablePolicyType
	organisasi: EnableAllFeatures
	Organisasi: aktifkan AWSServiceAccess
	organisasi: EnablePolicyType
	organisasi: InviteAccountToOrganization
	organisasi: LeaveOrganization
	organisasi: ListAccounts
	organisasi: ListAccountsForParent
	Organisasi: daftar AWSServiceAccessForOrganization
	organisasi: ListChildren
	organisasi: ListCreateAccountStatus
	organisasi: ListDelegatedAdministrators
	organisasi: ListDelegatedServicesForAccount
	organisasi: ListHandshakesForAccount
	organisasi: ListHandshakesForOrganization
	organisasi: ListOrganizationalUnitsForParent
	organisasi: ListParents

Awalan layanan	Tindakan
	organisasi: ListPolicies
	organisasi: ListPoliciesForTarget
	organisasi: ListRoots
	organisasi: ListTargetsForPolicy
	organisasi: MoveAccount
	organisasi: PutResourcePolicy
	organisasi: RegisterDelegatedAdministrator
	organisasi: RemoveAccountFromOrganization
	organisasi: UpdateOrganizationalUnit
	organisasi: UpdatePolicy

Awalan layanan	Tindakan
pos terdepan	pos terdepan: CancelOrder pos terdepan: CreateOrder pos terdepan: CreateOutpost pos terdepan: CreatePrivateConnectivityConfig pos terdepan: CreateSite pos terdepan: DeleteOutpost pos terdepan: DeleteSite pos terdepan: GetCatalogItem pos terdepan: GetConnection pos terdepan: GetOrder pos terdepan: GetOutpost pos terdepan: GetOutpostInstanceTypes pos terdepan: GetPrivateConnectivityConfig pos terdepan: GetSite pos terdepan: GetSiteAddress pos terdepan: ListAssets pos terdepan: ListCatalogItems pos terdepan: ListOrders pos terdepan: ListOutposts pos terdepan: ListSites pos terdepan: StartConnection

Awalan layanan	Tindakan
	pos terdepan: UpdateOutpost pos terdepan: UpdateSite pos terdepan: UpdateSiteAddress pos terdepan: UpdateSiteRackPhysicalProperties

Awalan layanan	Tindakan
panorama	panorama: CreateApplicationInstance panorama: CreateJobForDevices panorama: CreateNodeFromTemplateJob panorama: CreatePackage panorama: CreatePackageImportJob panorama: DeleteDevice panorama: DeletePackage panorama: DeregisterPackageVersion panorama: DescribeApplicationInstance panorama: DescribeApplicationInstanceDetails panorama: DescribeDevice panorama: DescribeDeviceJob panorama: DescribeNode panorama: DescribeNodeFromTemplateJob panorama: DescribePackage panorama: DescribePackageImportJob panorama: DescribePackageVersion panorama: ListApplicationInstanceDependencies panorama: ListApplicationInstanceNodeInstances panorama: ListApplicationInstances panorama: ListDevices

Awalan layanan	Tindakan
	<p>panorama: ListDevicesJobs</p> <p>panorama: ListNodeFromTemplateJobs</p> <p>panorama: ListNodes</p> <p>panorama: ListPackageImportJobs</p> <p>panorama: ListPackages</p> <p>panorama: ProvisionDevice</p> <p>panorama: RegisterPackageVersion</p> <p>panorama: RemoveApplicationInstance</p> <p>panorama: SignalApplicationInstanceNodeInstances</p> <p>panorama: UpdateDeviceMetadata</p>
pi	<p>pi: CreatePerformanceAnalysisReport</p> <p>pi: DeletePerformanceAnalysisReport</p> <p>pi: DescribeDimensionKeys</p> <p>pi: GetDimensionKeyDetails</p> <p>pi: GetPerformanceAnalysisReport</p> <p>pi: GetResourceMetadata</p> <p>pi: GetResourceMetrics</p> <p>pi: ListAvailableResourceDimensions</p> <p>pi: ListAvailableResourceMetrics</p> <p>pi: ListPerformanceAnalysisReports</p>

Awalan layanan	Tindakan
pipa	pipa: CreatePipe pipa: DeletePipe pipa: DescribePipe pipa: ListPipes pipa: StartPipe pipa: StopPipe pipa: UpdatePipe
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: CreateCalculatedAttributeDefinition Profil: CreateDomain Profil: CreateEventStream Profil: CreateProfile Profil: DeleteCalculatedAttributeDefinition Profil: DeleteDomain Profil: DeleteEventStream Profil: DeleteIntegration Profil: DeleteProfile Profil: DeleteProfileKey Profil: DeleteProfileObject Profil: DeleteProfileObjectType Profil: DeleteWorkflow Profil: DetectProfileObjectType Profil: GetAutoMergingPreview Profil: GetCalculatedAttributeDefinition Profil: GetCalculatedAttributeForProfile Profil: GetDomain Profil: GetEventStream Profil: GetIdentityResolutionJob

Awalan layanan	Tindakan
	Profil: GetIntegration
	Profil: GetMatches
	Profil: GetProfileObjectType
	Profil: GetProfileObjectTypeTemplate
	Profil: GetSimilarProfiles
	Profil: GetWorkflow
	Profil: GetWorkflowSteps
	Profil: ListAccountIntegrations
	Profil: ListCalculatedAttributeDefinitions
	Profil: ListCalculatedAttributesForProfile
	Profil: ListDomains
	Profil: ListEventStreams
	Profil: ListIdentityResolutionJobs
	Profil: ListIntegrations
	Profil: ListProfileObjects
	Profil: ListProfileObjectTypes
	Profil: ListProfileObjectTypeTemplates
	Profil: ListRuleBasedMatches
	Profil: ListWorkflows
	Profil: MergeProfiles
	Profil: PutIntegration

Awalan layanan	Tindakan
	Profil: PutProfileObject
	Profil: PutProfileObjectType
	Profil: SearchProfiles
	Profil: UpdateCalculatedAttributeDefinition
	Profil: UpdateDomain
	Profil: UpdateProfile

Awalan layanan	Tindakan
qldb	qldb: CancelJournalKinesisStream qldb: CreateLedger qldb: DeleteLedger qldb: DescribeJournalKinesisStream qldb: S3Ekspor DescribeJournal qldb: DescribeLedger qldb: S3 ExportJournalTo qldb: GetBlock qldb: GetDigest qldb: GetRevision qldb: ListJournalKinesisStreamsForLedger qldb: S3Ekspor ListJournal qldb: S3 ListJournal ExportsForLedger 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: ListPermissions ram: ListPermissionVersions

Awalan layanan	Tindakan
	ram: ListPrincipals ram: ListReplacePermissionAssociationsWork ram: ListResources ram: ListResourceSharePermissions ram: ListResourceTypes ram: PromotePermissionCreatedFromPolicy ram: PromoteResourceShareCreatedFromPolicy ram: RejectResourceShareInvitation ram: ReplacePermissionAssociations ram: SetDefaultPermissionVersion ram: UpdateResourceShare
rbin	rbin: CreateRule rbin: DeleteRule rbin: GetRule rbin: ListRules rbin: LockRule rbin: UnlockRule rbin: UpdateRule

Awalan layanan	Tindakan
rds	rds: DBCluster AddRoleTo rds: DBInstance AddRoleTo rds: AddSourceIdentifierToSubscription rds: ApplyPendingMaintenanceAction RDS: AuthorizedB SecurityGroupIngress RDS: BacktrackDBCluster rds: CancelExportTask RDS: CopyDB ClusterParameterGroup RDS: CopyDB ClusterSnapshot RDS: CopyDB ParameterGroup RDS: CopyDBSnapshot rds: CopyOptionGroup rds: DB CreateCustom EngineVersion RDS: dibuatDB ClusterParameterGroup RDS: dibuatDB ClusterSnapshot RDS: dibuatDB ParameterGroup RDS: dibuatBProxy RDS: dibuatDB ProxyEndpoint RDS: dibuatDB SecurityGroup RDS: dibuatBSnapshot RDS: dibuatDB SubnetGroup

Awalan layanan	Tindakan
	rds: CreateEventSubscription
	rds: CreateGlobalCluster
	rds: CreateOptionGroup
	rds: DeleteBlueGreenDeployment
	RDS: DihapusB ClusterAutomatedBackup
	RDS: DihapusB ClusterParameterGroup
	RDS: DihapusB ClusterSnapshot
	RDS: DihapusB InstanceAutomatedBackup
	RDS: DihapusB ParameterGroup
	RDS: DeletedBProxy
	RDS: DihapusB ProxyEndpoint
	RDS: DihapusB SecurityGroup
	RDS: DeletedBSnapshot
	RDS: DihapusB SubnetGroup
	rds: DeleteEventSubscription
	rds: DeleteGlobalCluster
	rds: DeleteOptionGroup
	RDS: DeregisterDB ProxyTargets
	rds: DescribeAccountAttributes
	rds: DescribeBlueGreenDeployments
	rds: DescribeCertificates

Awalan layanan	Tindakan
	RDS: dijelaskanB ClusterAutomatedBackups
	RDS: dijelaskanB ClusterBacktracks
	RDS: dijelaskanB ClusterEndpoints
	RDS: dijelaskanB ClusterParameterGroups
	RDS: dijelaskanB ClusterParameters
	RDS: DescribedBClusters
	RDS: dijelaskanB ClusterSnapshotAttributes
	RDS: dijelaskanB ClusterSnapshots
	RDS: dijelaskanB EngineVersions
	RDS: dijelaskanB InstanceAutomatedBackups
	RDS: DescribedBInstances
	RDS: dijelaskanB LogFiles
	RDS: dijelaskanB ParameterGroups
	RDS: DescribedBParameters
	RDS: DescribedBProxies
	RDS: dijelaskanB ProxyEndpoints
	RDS: dijelaskanB ProxyTargetGroups
	RDS: dijelaskanB ProxyTargets
	RDS:DescribedBrecommendations
	RDS: dijelaskanB SecurityGroups
	RDS: dijelaskanB SnapshotAttributes

Awalan layanan	Tindakan
	RDS: DescribeDBSnapshots
	rds: DescribeDbSnapshotTenantDatabases
	RDS: dijelaskanB SubnetGroups
	rds: DescribeEngineDefaultClusterParameters
	rds: DescribeEngineDefaultParameters
	rds: DescribeEventCategories
	rds: DescribeEvents
	rds: DescribeEventSubscriptions
	rds: DescribeExportTasks
	rds: DescribeGlobalClusters
	rds: DescribeIntegrations
	rds: DescribeOptionGroupOptions
	rds: DescribeOptionGroups
	rds: DB DescribeOrderable InstanceOptions
	rds: DescribePendingMaintenanceActions
	rds: dbInstances DescribeReserved
	rds: DB DescribeReserved InstancesOfferings
	rds: DescribeSourceRegions
	rds: DescribeTenantDatabases
	rds: DB DescribeValid InstanceModifications
	rds: DB DownloadComplete LogFile

Awalan layanan	Tindakan
	RDS: DownloAdDB LogFilePortion
	RDS: FailoverDBCluster
	rds: FailoverGlobalCluster
	rds: ModifyActivityStream
	rds: ModifyCertificates
	rds: DB ModifyCurrent ClusterCapacity
	RDS: ModifyDB ClusterEndpoint
	RDS: ModifyDB ClusterParameterGroup
	RDS: ModifyDB ClusterSnapshotAttribute
	RDS: ModifyDB ParameterGroup
	RDS: ModifyDBProxy
	RDS: ModifyDB ProxyEndpoint
	RDS: ModifyDB ProxyTargetGroup
	RDS: ModifydbreCommendation
	RDS: ModifyDBSnapshot
	RDS: ModifyDB SnapshotAttribute
	RDS: ModifyDB SubnetGroup
	rds: ModifyEventSubscription
	rds: ModifyGlobalCluster
	rds: ModifyOptionGroup
	rds: ModifyTenantDatabase

Awalan layanan	Tindakan
	rds: DBPurchaseReservedInstancesOffering
	RDS: RebootDBCluster
	RDS: DaftarDBProxyTargets
	rds: RemoveFromGlobalCluster
	rds: DBCluster RemoveRoleFrom
	rds: DBInstance RemoveRoleFrom
	rds: RemoveSourceIdentifierFromSubscription
	RDS: ResetDBClusterParameterGroup
	RDS: ResetDBParameterGroup
	RDSClusterFrom: DipulihkanB S3
	RDS: DipulihkanB ClusterFromSnapshot
	RDS: DikembalikanB ClusterToPointInTime
	RDS: DipulihkanB DBSnapshot InstanceFrom
	RDSInstanceFrom: DipulihkanB S3
	RDS: DikembalikanB InstanceToPointInTime
	RDS: dicabut SecurityGroupIngress
	rds: StartActivityStream
	RDS: StartDBCluster
	RDS: StartDBInstance
	RDS: StartDB InstanceAutomatedBackupsReplication
	rds: StartExportTask

Awalan layanan	Tindakan
	rds: StopActivityStream RDS: STOPDBCluster RDS: StopdbInstance RDS: StopDB InstanceAutomatedBackupsReplication 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: 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: DeleteScheduledAction
	pergeseran merah: DeleteSnapshotCopyGrant

Awalan layanan	Tindakan
	pergeseran merah: DeleteSnapshotSchedule
	pergeseran merah: DeleteUsageLimit
	pergeseran merah: DescribeAccountAttributes
	pergeseran merah: DescribeAuthenticationProfiles
	pergeseran merah: DescribeClusterDbRevisions
	pergeseran merah: DescribeClusterParameterGroups
	pergeseran merah: DescribeClusterParameters
	pergeseran merah: DescribeClusters
	pergeseran merah: DescribeClusterSecurityGroups
	pergeseran merah: DescribeClusterSnapshots
	pergeseran merah: DescribeClusterSubnetGroups
	pergeseran merah: DescribeClusterTracks
	pergeseran merah: DescribeClusterVersions
	pergeseran merah: DescribeCustomDomainAssociations
	pergeseran merah: DescribeDataShares
	pergeseran merah: DescribeDataSharesForConsumer
	pergeseran merah: DescribeDataSharesForProducer
	pergeseran merah: DescribeDefaultClusterParameters
	pergeseran merah: DescribeEndpointAccess
	pergeseran merah: DescribeEndpointAuthorization
	pergeseran merah: DescribeEventCategories

Awalan layanan	Tindakan
	<p>pergeseran merah: DescribeEvents</p> <p>pergeseran merah: DescribeEventSubscriptions</p> <p>pergeseran merah: DescribeHsmClientCertificates</p> <p>pergeseran merah: DescribeHsmConfigurations</p> <p>pergeseran merah: DescribeInboundIntegrations</p> <p>pergeseran merah: DescribeLoggingStatus</p> <p>pergeseran merah: DescribeNodeConfigurationOptions</p> <p>pergeseran merah: DescribeOrderableClusterOptions</p> <p>pergeseran merah: DescribePartners</p> <p>pergeseran merah: DescribeRedshiftIdcApplications</p> <p>pergeseran merah: DescribeReservedNodeExchangeStatus</p> <p>pergeseran merah: DescribeReservedNodeOfferings</p> <p>pergeseran merah: DescribeReservedNodes</p> <p>pergeseran merah: DescribeResize</p> <p>pergeseran merah: DescribeScheduledActions</p> <p>pergeseran merah: DescribeSnapshotCopyGrants</p> <p>pergeseran merah: DescribeSnapshotSchedules</p> <p>pergeseran merah: DescribeStorage</p> <p>pergeseran merah: DescribeTableRestoreStatus</p> <p>pergeseran merah: DescribeUsageLimits</p> <p>pergeseran merah: DisableLogging</p>

Awalan layanan	Tindakan
	<p>pergeseran merah: DisableSnapshotCopy</p> <p>pergeseran merah: DisassociateDataShareConsumer</p> <p>pergeseran merah: EnableLogging</p> <p>pergeseran merah: EnableSnapshotCopy</p> <p>pergeseran merah: FailoverPrimaryCompute</p> <p>pergeseran merah: GetClusterCredentials</p> <p>pergeseran merah: GetClusterCredentialsWith IAM</p> <p>pergeseran merah: GetReservedNodeExchangeConfiguration Options</p> <p>pergeseran merah: GetReservedNodeExchangeOfferings</p> <p>pergeseran merah: ListRecommendations</p> <p>pergeseran merah: ModifyAquaConfiguration</p> <p>pergeseran merah: ModifyAuthenticationProfile</p> <p>pergeseran merah: ModifyCluster</p> <p>pergeseran merah: ModifyClusterDbRevision</p> <p>pergeseran merah: ModifyClusterIamRoles</p> <p>pergeseran merah: ModifyClusterMaintenance</p> <p>pergeseran merah: ModifyClusterParameterGroup</p> <p>pergeseran merah: ModifyClusterSnapshot</p> <p>pergeseran merah: ModifyClusterSnapshotSchedule</p> <p>pergeseran merah: ModifyClusterSubnetGroup</p>

Awalan layanan	Tindakan
	pergeseran merah: ModifyCustomDomainAssociation
	pergeseran merah: ModifyEndpointAccess
	pergeseran merah: ModifyEventSubscription
	pergeseran merah: ModifyScheduledAction
	pergeseran merah: ModifySnapshotCopyRetentionPeriod
	pergeseran merah: ModifySnapshotSchedule
	pergeseran merah: ModifyUsageLimit
	pergeseran merah: PauseCluster
	pergeseran merah: PurchaseReservedNodeOffering
	pergeseran merah: RebootCluster
	pergeseran merah: RejectDataShare
	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

Awalan layanan	Tindakan
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: 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: ListEnvironments ruang refaktor: ListEnvironmentVpcs 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: DescribeProjects

Awalan layanan	Tindakan
	Rekognisi: DescribeProjectVersions
	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: AddDraftAppVersionResourceMappings 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 resiliencehub: DescribeAppVersionTemplate resiliencehub: DescribeDraftAppVersionResourcesImportStatus

Awalan layanan	Tindakan
	<p>resiliencehub: DescribeResiliencyPolicy</p> <p>resiliencehub: ImportResourcesToDraftAppVersion</p> <p>resiliencehub: ListAlarmRecommendations</p> <p>resiliencehub: ListAppAssessments</p> <p>resiliencehub: ListAppComponentCompliances</p> <p>resiliencehub: ListAppComponentRecommendations</p> <p>resiliencehub: ListAppInputSources</p> <p>resiliencehub: ListApps</p> <p>resiliencehub: ListAppVersionAppComponents</p> <p>resiliencehub: ListAppVersionResourceMappings</p> <p>resiliencehub: ListAppVersionResources</p> <p>resiliencehub: ListAppVersions</p> <p>resiliencehub: ListRecommendationTemplates</p> <p>resiliencehub: ListResiliencyPolicies</p> <p>resiliencehub: ListSopRecommendations</p> <p>resiliencehub: ListSuggestedResiliencyPolicies</p> <p>resiliencehub: ListTestRecommendations</p> <p>resiliencehub: ListUnsupportedAppVersionResources</p> <p>resiliencehub: PublishAppVersion</p> <p>resiliencehub: PutDraftAppVersionTemplate</p> <p>resiliencehub: RemoveDraftAppVersionResourceMappings</p>

Awalan layanan	Tindakan
	<ul style="list-style-type: none">resiliencehub: ResolveAppVersionResourcesresiliencehub: StartAppAssessmentresiliencehub: UpdateAppresiliencehub: UpdateAppVersionresiliencehub: UpdateAppVersionAppComponentresiliencehub: UpdateAppVersionResourceresiliencehub: 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: CreateView
	sumber daya-penjelajah-2: DeleteIndex
	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: ListIndexes
	sumber daya-penjelajah-2: ListIndexesForMembers
	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: CreateGroup kelompok sumber daya: DeleteGroup kelompok sumber daya: GetAccountSettings kelompok sumber daya: GetGroup kelompok sumber daya: GetGroupConfiguration kelompok sumber daya: GetGroupQuery kelompok sumber daya: GroupResources kelompok sumber daya: ListGroupResources kelompok sumber daya: ListGroups kelompok sumber daya: PutGroupConfiguration kelompok sumber daya: SearchResources kelompok sumber daya: UngroupResources kelompok sumber daya: UpdateAccountSettings kelompok sumber daya: UpdateGroup kelompok sumber daya: UpdateGroupQuery

Awalan layanan	Tindakan
pembuat robomak	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: ListWorlds
	pembuat robomak: ListWorldTemplates
	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: 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 peranandi mana saja: PutNotificationSettings

Awalan layanan	Tindakan
	peranandi mana saja: ResetNotificationSettings peranandi mana saja: UpdateCrl peranandi mana saja: UpdateProfile peranandi mana saja: UpdateTrustAnchor

Awalan layanan	Tindakan
route53	route53: ActivateKeySigningKey Route53:AssociateVPC WithHostedZone route53: ChangeCidrCollection route53: ChangeResourceRecordSets route53: CreateCidrCollection route53: CreateHealthCheck route53: CreateHostedZone route53: CreateKeySigningKey route53: CreateQueryLoggingConfig route53: CreateReusableDelegationSet route53: CreateTrafficPolicy route53: CreateTrafficPolicyInstance route53: CreateTrafficPolicyVersion Route53:CreateVPC AssociationAuthorization route53: DeactivateKeySigningKey route53: DeleteCidrCollection route53: DeleteHealthCheck route53: DeleteHostedZone route53: DeleteKeySigningKey route53: DeleteQueryLoggingConfig route53: DeleteReusableDelegationSet

Awalan layanan	Tindakan
	route53: DeleteTrafficPolicy
	route53: DeleteTrafficPolicyInstance
	Route53:deleteVPC AssociationAuthorization
	route53: DNSSEC DisableHostedZone
	Route53:DisassociateVPC FromHostedZone
	route53: DNSSEC EnableHostedZone
	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: VPC ListHostedZonesBy
	route53: ListQueryLoggingConfigs
	route53: ListResourceRecordSets
	route53: ListReusableDelegationSets
	route53: ListTrafficPolicies
	route53: ListTrafficPolicyInstances
	route53: ListTrafficPolicyInstancesByHostedZone
	route53: ListTrafficPolicyInstancesByPolicy
	route53: ListTrafficPolicyVersions
	Route53: ListVPC AssociationAuthorizations
	Route53:testDNSsAnswer

Awalan layanan	Tindakan
	route53: UpdateHealthCheck
	route53: UpdateHostedZoneComment
	route53: UpdateTrafficPolicyComment
	route53: UpdateTrafficPolicyInstance

Awalan layanan	Tindakan
route53- recovery-control-config	<p>route53-: recovery-control-config CreateCluster</p> <p>route53-: recovery-control-config CreateControlPanel</p> <p>route53-: recovery-control-config CreateRoutingControl</p> <p>route53-: recovery-control-config CreateSafetyRule</p> <p>route53-: recovery-control-config DeleteCluster</p> <p>route53-: recovery-control-config DeleteControlPanel</p> <p>route53-: recovery-control-config DeleteRoutingControl</p> <p>route53-: recovery-control-config DeleteSafetyRule</p> <p>route53-: recovery-control-config DescribeCluster</p> <p>route53-: recovery-control-config DescribeControlPanel</p> <p>route53-: recovery-control-config DescribeRoutingControl</p> <p>route53-: recovery-control-config DescribeSafetyRule</p> <p>route53-: recovery-control-config GetResourcePolicy</p> <p>route53-: recovery-control-config 53 ListAssociatedRoute HealthChecks</p> <p>route53-: recovery-control-config ListClusters</p> <p>route53-: recovery-control-config ListControlPanels</p> <p>route53-: recovery-control-config ListRoutingControls</p> <p>route53-: recovery-control-config ListSafetyRules</p> <p>route53-: recovery-control-config UpdateControlPanel</p> <p>route53-: recovery-control-config UpdateRoutingControl</p>

Awalan layanan	Tindakan
	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
	<p>route53resolver: GetFirewallConfig</p> <p>route53resolver: GetFirewallDomainList</p> <p>route53resolver: GetFirewallRuleGroup</p> <p>route53resolver: GetFirewallRuleGroupAssociation</p> <p>route53resolver: GetFirewallRuleGroupPolicy</p> <p>route53resolver: GetOutpostResolver</p> <p>route53resolver: GetResolverConfig</p> <p>route53resolver: GetResolverDnssecConfig</p> <p>route53resolver: GetResolverEndpoint</p> <p>route53resolver: GetResolverQueryLogConfig</p> <p>route53resolver: GetResolverQueryLogConfigAssociation</p> <p>route53resolver: GetResolverQueryLogConfigPolicy</p> <p>route53resolver: GetResolverRule</p> <p>route53resolver: GetResolverRuleAssociation</p> <p>route53resolver: GetResolverRulePolicy</p> <p>route53resolver: ImportFirewallDomains</p> <p>route53resolver: ListFirewallConfigs</p> <p>route53resolver: ListFirewallDomainLists</p> <p>route53resolver: ListFirewallDomains</p> <p>route53resolver: ListFirewallRuleGroupAssociations</p> <p>route53resolver: ListFirewallRuleGroups</p>

Awalan layanan	Tindakan
	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
	route53resolver: UpdateResolverRule

Awalan layanan	Tindakan
rum	rum: BatchCreateRumMetricDefinitions rum: BatchDeleteRumMetricDefinitions rum: BatchGetRumMetricDefinitions rum: CreateAppMonitor rum: DeleteAppMonitor rum: DeleteRumMetricsDestination rum: GetAppMonitor rum: GetAppMonitorData rum: ListAppMonitors rum: ListRumMetricsDestinations 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: CreateJob s3: CreateMultiRegionAccessPoint s3: DeleteAccessGrant s3: DeleteAccessGrantsInstance s3: DeleteAccessGrantsInstanceResourcePolicy s3: DeleteAccessGrantsLocation s3: DeleteAccessPoint s3: DeleteAccessPointForObjectLambda s3: DeleteAccessPointPolicy s3: DeleteAccessPointPolicyForObjectLambda s3: PutAccountPublicAccessBlock s3: DeleteBucket s3: PutAnalyticsConfiguration s3: PutBucket CORS

Awalan layanan	Tindakan
	<p>s3: PutEncryptionConfiguration</p> <p>s3: PutIntelligentTieringConfiguration</p> <p>s3: PutInventoryConfiguration</p> <p>s3: PutLifecycleConfiguration</p> <p>s3: PutMetricsConfiguration</p> <p>s3: PutBucketOwnershipControls</p> <p>s3: DeleteBucketPolicy</p> <p>s3: PutBucketPublicAccessBlock</p> <p>s3: PutReplicationConfiguration</p> <p>s3: DeleteBucketWebsite</p> <p>s3: DeleteMultiRegionAccessPoint</p> <p>s3: DeleteStorageLensConfiguration</p> <p>s3: DescribeJob</p> <p>s3: DescribeMultiRegionAccessPointOperation</p> <p>s3: DissociateAccessGrantsIdentityCenter</p> <p>s3: GetAccelerateConfiguration</p> <p>s3: GetAccessGrant</p> <p>s3: GetAccessGrantsInstance</p> <p>s3: GetAccessGrantsInstanceForPrefix</p> <p>s3: GetAccessGrantsInstanceResourcePolicy</p> <p>s3: GetAccessGrantsLocation</p>

Awalan layanan	Tindakan
	s3: GetAccessPoint
	s3: GetAccessPointConfigurationForObjectLambda
	s3: GetAccessPointForObjectLambda
	s3: GetAccessPointPolicy
	s3: GetAccessPointPolicyForObjectLambda
	s3: GetAccessPointPolicyStatus
	s3: GetAccessPointPolicyStatusForObjectLambda
	s3: GetAccountPublicAccessBlock
	s3: GetBucketAcl
	s3: GetAnalyticsConfiguration
	s3: GetBucket CORS
	s3: GetEncryptionConfiguration
	s3: GetIntelligentTieringConfiguration
	s3: GetInventoryConfiguration
	s3: GetLifecycleConfiguration
	s3: GetBucketLocation
	s3: GetBucketLogging
	s3: GetMetricsConfiguration
	s3: GetBucketNotification
	s3: GetBucketObjectLockConfiguration
	s3: GetBucketOwnershipControls

Awalan layanan	Tindakan
	<ul style="list-style-type: none">s3: GetBucketPolicys3: GetBucketPolicyStatuss3: GetBucketPublicAccessBlocks3: GetReplicationConfigurations3: GetBucketRequestPayments3: GetBucketVersionings3: GetBucketWebsites3: GetDataAccesss3: GetMultiRegionAccessPoints3: GetMultiRegionAccessPointPolicys3: GetMultiRegionAccessPointPolicyStatuss3: GetMultiRegionAccessPointRoutess3: GetObjectAttributess3: GetStorageLensConfigurations3: GetStorageLensDashboards3: ListAccessGrantss3: ListAccessGrantsInstancess3: ListAccessGrantsLocationss3: ListAccessPointss3: ListAccessPointsForObjectLambdas3: ListAllMyBuckets

Awalan layanan	Tindakan
	<ul style="list-style-type: none">s3: ListJobss3: ListBucketMultipartUploadss3: ListMultiRegionAccessPointss3: ListStorageLensConfigurationss3: PutAccelerateConfigurations3: PutAccessGrantsInstanceResourcePolicys3: PutAccessPointConfigurationForObjectLambdas3: PutAccessPointPolicys3: PutAccessPointPolicyForObjectLambdas3: PutAccountPublicAccessBlocks3: PutBucketAcls3: PutAnalyticsConfigurations3: PutBucket CORSs3: PutEncryptionConfigurations3: PutIntelligentTieringConfigurations3: PutInventoryConfigurations3: PutLifecycleConfigurations3: PutBucketLoggings3: PutMetricsConfigurations3: PutBucketNotifications3: PutBucketObjectLockConfiguration

Awalan layanan	Tindakan
	<p>s3: PutBucketOwnershipControls</p> <p>s3: PutBucketPolicy</p> <p>s3: PutBucketPublicAccessBlock</p> <p>s3: PutReplicationConfiguration</p> <p>s3: PutBucketRequestPayment</p> <p>s3: PutBucketVersioning</p> <p>s3: PutBucketWebsite</p> <p>s3: PutMultiRegionAccessPointPolicy</p> <p>s3: PutStorageLensConfiguration</p> <p>s3: SubmitMultiRegionAccessPointRoutes</p> <p>s3: UpdateAccessGrantsLocation</p> <p>s3: UpdateJobPriority</p> <p>s3: UpdateJobStatus</p>
pos terdepan s3	<p>pos terdepan s3: CreateEndpoint</p> <p>pos terdepan s3: DeleteEndpoint</p> <p>pos terdepan s3: ListEndpoints</p> <p>s3-pos terdepan: S3 ListOutpostsWith</p> <p>pos terdepan s3: ListSharedEndpoints</p>

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: ListSchemas skema: ListSchemaVersions 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: ListSecrets manajer rahasia: ListSecretVersionIds 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: BatchGetSecurityControls securityhub: BatchGetStandardsControlAssociations securityhub: BatchImportFindings securityhub: BatchUpdateAutomationRules securityhub: BatchUpdateFindings securityhub: BatchUpdateStandardsControlAssociations securityhub: CreateActionTarget securityhub: CreateAutomationRule securityhub: CreateFindingAggregator securityhub: CreateInsight securityhub: CreateMembers securityhub: DeclineInvitations securityhub: DeleteActionTarget securityhub: DeleteFindingAggregator securityhub: DeleteInsight

Awalan layanan	Tindakan
	securityhub: DeleteInvitations
	securityhub: DeleteMembers
	securityhub: DescribeActionTargets
	securityhub: DescribeHub
	securityhub: DescribeOrganizationConfiguration
	securityhub: DescribeProducts
	securityhub: DescribeStandards
	securityhub: DisableImportFindingsForProduct
	securityhub: DisableOrganizationAdminAccount
	securityhub: DisableSecurityHub
	securityhub: DisassociateFromAdministratorAccount
	securityhub: DisassociateFromMasterAccount
	securityhub: DisassociateMembers
	securityhub: EnableImportFindingsForProduct
	securityhub: EnableOrganizationAdminAccount
	securityhub: EnableSecurityHub
	securityhub: GetAdministratorAccount
	securityhub: GetEnabledStandards
	securityhub: GetFindingAggregator
	securityhub: GetFindingHistory
	securityhub: GetFindings

Awalan layanan	Tindakan
	securityhub: GetInsightResults
	securityhub: GetInsights
	securityhub: GetInvitationsCount
	securityhub: GetMasterAccount
	securityhub: GetMembers
	securityhub: GetSecurityControlDefinition
	securityhub: InviteMembers
	securityhub: ListAutomationRules
	securityhub: ListEnabledProductsForImport
	securityhub: ListFindingAggregators
	securityhub: ListInvitations
	securityhub: ListMembers
	securityhub: ListOrganizationAdminAccounts
	securityhub: ListSecurityControlDefinitions
	securityhub: ListStandardsControlAssociations
	securityhub: StartConfigurationPolicyDisassociation
	securityhub: UpdateActionTarget
	securityhub: UpdateFindingAggregator
	securityhub: UpdateFindings
	securityhub: UpdateInsight
	securityhub: UpdateOrganizationConfiguration

Awalan layanan	Tindakan
	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: ListApplications repo tanpa server: ListApplicationVersions 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: CreateConstraint servicecatalog: CreatePortfolio servicecatalog: CreatePortfolioShare servicecatalog: CreateProduct servicecatalog: CreateProvisionedProductPlan servicecatalog: CreateProvisioningArtifact servicecatalog: CreateServiceAction servicecatalog: DeleteConstraint servicecatalog: DeletePortfolio servicecatalog: DeletePortfolioShare servicecatalog: DeleteProduct servicecatalog: DeleteProvisionedProductPlan

Awalan layanan	Tindakan
	<p>servicecatalog: DeleteProvisioningArtifact</p> <p>servicecatalog: DeleteServiceAction</p> <p>servicecatalog: DescribeConstraint</p> <p>servicecatalog: DescribeCopyProductStatus</p> <p>servicecatalog: DescribePortfolio</p> <p>servicecatalog: DescribePortfolioShares</p> <p>servicecatalog: DescribePortfolioShareStatus</p> <p>servicecatalog: DescribeProduct</p> <p>servicecatalog: DescribeProductAsAdmin</p> <p>servicecatalog: DescribeProductView</p> <p>servicecatalog: DescribeProvisionedProductPlan</p> <p>servicecatalog: DescribeProvisioningArtifact</p> <p>servicecatalog: DescribeProvisioningParameters</p> <p>servicecatalog: DescribeRecord</p> <p>servicecatalog: DescribeServiceAction</p> <p>servicecatalog: DescribeServiceActionExecutionParameters</p> <p>ServiceCatalog:Nonaktifkan AWSOrganizationsAccess</p> <p>servicecatalog: DisassociateBudgetFromResource</p> <p>servicecatalog: DisassociatePrincipalFromPortfolio</p> <p>servicecatalog: DisassociateProductFromPortfolio</p> <p>servicecatalog: DisassociateServiceActionFromProvisioningArtifact</p>

Awalan layanan	Tindakan
	<p>ServiceCatalog:Aktifkan AWSOrganizationsAccess</p> <p>servicecatalog: ExecuteProvisionedProductPlan</p> <p>servicecatalog: ExecuteProvisionedProductServiceAction</p> <p>ServiceCatalog:Dapatkan AWSOrganizationsAccessStatus</p> <p>servicecatalog: GetProvisionedProductOutputs</p> <p>servicecatalog: ImportAsProvisionedProduct</p> <p>servicecatalog: ListAcceptedPortfolioShares</p> <p>servicecatalog: ListBudgetsForResource</p> <p>servicecatalog: ListConstraintsForPortfolio</p> <p>servicecatalog: ListLaunchPaths</p> <p>servicecatalog: ListOrganizationPortfolioAccess</p> <p>servicecatalog: ListPortfolioAccess</p> <p>servicecatalog: ListPortfolios</p> <p>servicecatalog: ListPortfoliosForProduct</p> <p>servicecatalog: ListPrincipalsForPortfolio</p> <p>servicecatalog: ListProvisionedProductPlans</p> <p>servicecatalog: ListProvisioningArtifacts</p> <p>servicecatalog: ListProvisioningArtifactsForServiceAction</p> <p>servicecatalog: ListRecordHistory</p> <p>servicecatalog: ListServiceActions</p> <p>servicecatalog: ListServiceActionsForProvisioningArtifact</p>

Awalan layanan	Tindakan
	<p>servicecatalog: ListStackInstancesForProvisionedProduct</p> <p>servicecatalog: NotifyProvisionProductEngineWorkflowResult</p> <p>servicecatalog: NotifyTerminateProvisionedProductEngineWorkflowResult</p> <p>servicecatalog: NotifyUpdateProvisionedProductEngineWorkflowResult</p> <p>servicecatalog: ProvisionProduct</p> <p>servicecatalog: RejectPortfolioShare</p> <p>servicecatalog: ScanProvisionedProducts</p> <p>servicecatalog: SearchProducts</p> <p>servicecatalog: SearchProductsAsAdmin</p> <p>servicecatalog: SearchProvisionedProducts</p> <p>servicecatalog: TerminateProvisionedProduct</p> <p>servicecatalog: UpdateConstraint</p> <p>servicecatalog: UpdatePortfolio</p> <p>servicecatalog: UpdatePortfolioShare</p> <p>servicecatalog: UpdateProduct</p> <p>servicecatalog: UpdateProvisionedProduct</p> <p>servicecatalog: UpdateProvisionedProductProperties</p> <p>servicecatalog: UpdateProvisioningArtifact</p> <p>servicecatalog: UpdateServiceAction</p>

Awalan layanan	Tindakan
servicediscovery	penemuan layanan: CreateHttpNamespace penemuan layanan: CreatePrivateDnsNamespace penemuan layanan: CreatePublicDnsNamespace penemuan layanan: CreateService penemuan layanan: DeleteNamespace penemuan layanan: DeleteService 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 penemuan layanan: UpdatePublicDnsNamespace

Awalan layanan	Tindakan
	penemuan layanan: UpdateService
servicequotas	servicequotas: AssociateServiceQuotaTemplate servicequotas: DeleteServiceQuotaIncreaseRequestFromTemplate servicequotas: DisassociateServiceQuotaTemplate servicequotas: GetAssociationForServiceQuotaTemplate ServiceQuotas:Dapatkan AWSDefaultServiceQuota servicequotas: GetRequestedServiceQuotaChange servicequotas: GetServiceQuota servicequotas: GetServiceQuotaIncreaseRequestFromTemplate ServiceQuotas:Daftar AWSDefaultServiceQuotas servicequotas: ListRequestedServiceQuotaChangeHistory servicequotas: ListRequestedServiceQuotaChangeHistoryByQuota servicequotas: ListServiceQuotaIncreaseRequestsInTemplate servicequotas: ListServiceQuotas servicequotas: ListServices servicequotas: PutServiceQuotaIncreaseRequestIntoTemplate servicequotas: RequestServiceQuotaIncrease

Awalan layanan	Tindakan
ses	ses: BatchGetMetricData ses: CloneReceiptRuleSet 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: CreateReceiptFilter ses: CreateReceiptRule ses: CreateReceiptRuleSet ses: CreateTemplate ses: DeleteConfigurationSet ses: DeleteConfigurationSetEventDestination ses: DeleteConfigurationSetTrackingOptions

Awalan layanan	Tindakan
	ses: DeleteContact
	ses: DeleteContactList
	ses: DeleteCustomVerificationEmailTemplate
	ses: DeleteDedicatedIpPool
	ses: DeleteEmailIdentity
	ses: DeleteEmailIdentityPolicy
	ses: DeleteEmailTemplate
	ses: DeleteIdentity
	ses: DeleteIdentityPolicy
	ses: DeleteReceiptFilter
	ses: DeleteReceiptRule
	ses: DeleteReceiptRuleSet
	ses: DeleteSuppressedDestination
	ses: DeleteTemplate
	ses: DeleteVerifiedEmailAddress
	ses: DescribeActiveReceiptRuleSet
	ses: DescribeConfigurationSet
	ses: DescribeReceiptRule
	ses: DescribeReceiptRuleSet
	ses: GetAccount
	ses: GetAccountSendingEnabled

Awalan layanan	Tindakan
	ses: GetBlacklistReports
	ses: GetConfigurationSet
	ses: GetConfigurationSetEventDestinations
	ses: GetContact
	ses: GetContactList
	ses: GetCustomVerificationEmailTemplate
	ses: GetDedicatedIp
	ses: GetDedicatedIpPool
	ses: GetDedicatedIps
	ses: GetDeliverabilityDashboardOptions
	ses: GetDeliverabilityTestReport
	ses: GetDomainDeliverabilityCampaign
	ses: GetDomainStatisticsReport
	ses: GetEmailIdentity
	ses: GetEmailIdentityPolicies
	ses: GetEmailTemplate
	ses: GetIdentityDkimAttributes
	ses: GetIdentityMailFromDomainAttributes
	ses: GetIdentityNotificationAttributes
	ses: GetIdentityPolicies
	ses: GetIdentityVerificationAttributes

Awalan layanan	Tindakan
	ses: GetImportJob
	ses: GetMessageInsights
	ses: GetSendQuota
	ses: GetSendStatistics
	ses: GetSuppressedDestination
	ses: GetTemplate
	ses: ListConfigurationSets
	ses: ListContactLists
	ses: ListContacts
	ses: ListCustomVerificationEmailTemplates
	ses: ListDedicatedIpPools
	ses: ListDeliverabilityTestReports
	ses: ListDomainDeliverabilityCampaigns
	ses: ListEmailIdentities
	ses: ListEmailTemplates
	ses: ListExportJobs
	ses: ListIdentities
	ses: ListIdentityPolicies
	ses: ListImportJobs
	ses: ListReceiptFilters
	ses: ListReceiptRuleSets

Awalan layanan	Tindakan
	ses: ListRecommendations
	ses: ListSuppressedDestinations
	ses: ListTemplates
	ses: ListVerifiedEmailAddresses
	ses: PutAccountDedicatedIpWarmupAttributes
	ses: PutAccountDetails
	ses: PutAccountSendingAttributes
	ses: PutAccountSuppressionAttributes
	ses: PutAccountVdmAttributes
	ses: PutConfigurationSetDeliveryOptions
	ses: PutConfigurationSetReputationOptions
	ses: PutConfigurationSetSendingOptions
	ses: PutConfigurationSetSuppressionOptions
	ses: PutConfigurationSetTrackingOptions
	ses: PutConfigurationSetVdmOptions
	ses: PutDedicatedIpInPool
	ses: PutDedicatedIpPoolScalingAttributes
	ses: PutDedicatedIpWarmupAttributes
	ses: PutDeliverabilityDashboardOption
	ses: PutEmailIdentityConfigurationSetAttributes
	ses: PutEmailIdentityDkimAttributes

Awalan layanan	Tindakan
	ses: PutEmailIdentityDkimSigningAttributes
	ses: PutEmailIdentityFeedbackAttributes
	ses: PutEmailIdentityMailFromAttributes
	ses: PutIdentityPolicy
	ses: PutSuppressedDestination
	ses: ReorderReceiptRuleSet
	ses: SendBounce
	ses: SendCustomVerificationEmail
	ses: SetActiveReceiptRuleSet
	ses: SetIdentityDkimEnabled
	ses: SetIdentityFeedbackForwardingEnabled
	ses: SetIdentityHeadersInNotificationsEnabled
	ses: SetIdentityMailFromDomain
	ses: SetIdentityNotificationTopic
	ses: SetReceiptRulePosition
	ses: TestRenderEmailTemplate
	ses: TestRenderTemplate
	ses: UpdateAccountSendingEnabled
	ses: UpdateConfigurationSetEventDestination
	ses: UpdateConfigurationSetReputationMetricsEnabled
	ses: UpdateConfigurationSetSendingEnabled

Awalan layanan	Tindakan
	ses: UpdateConfigurationSetTrackingOptions
	ses: UpdateContact
	ses: UpdateContactList
	ses: UpdateCustomVerificationEmailTemplate
	ses: UpdateEmailIdentityPolicy
	ses: UpdateEmailTemplate
	ses: UpdateReceiptRule
	ses: UpdateTemplate
	ses: VerifyDomainDkim
	ses: VerifyDomainIdentity
	ses: VerifyEmailAddress
	ses: VerifyEmailIdentity

Awalan layanan	Tindakan
melindungi	Perisai: AssociateDRT LogBucket 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: DisassociateDRT LogBucket Perisai: DisassociateDrTrole perisai: DisassociateHealthCheck

Awalan layanan	Tindakan
	<p>perisai: EnableApplicationLayerAutomaticResponse</p> <p>perisai: EnableProactiveEngagement</p> <p>perisai: GetSubscriptionState</p> <p>perisai: ListAttacks</p> <p>perisai: ListProtectionGroups</p> <p>perisai: ListProtections</p> <p>perisai: ListResourcesInProtectionGroup</p> <p>perisai: UpdateApplicationLayerAutomaticResponse</p> <p>perisai: UpdateEmergencyContactSettings</p> <p>perisai: UpdateProtectionGroup</p> <p>perisai: UpdateSubscription</p>

Awalan layanan	Tindakan
penandatanganan	penandatanganan: AddProfilePermission penandatanganan: CancelSigningProfile penandatanganan: DescribeSigningJob penandatanganan: GetRevocationStatus penandatanganan: GetSigningPlatform penandatanganan: GetSigningProfile penandatanganan: ListProfilePermissions penandatanganan: ListSigningJobs penandatanganan: ListSigningPlatforms penandatanganan: ListSigningProfiles penandatanganan: PutSigningProfile penandatanganan: RemoveProfilePermission penandatanganan: RevokeSignature penandatanganan: RevokeSigningProfile penandatanganan: SignPayload penandatanganan: 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: CreateConfigurationSet sms-suara: CreateConfigurationSetEventDestination sms-suara: CreateEventDestination sms-suara: CreateOptOutList sms-suara: CreatePool sms-suara: CreateRegistration sms-suara: CreateRegistrationAssociation sms-suara: CreateRegistrationAttachment sms-suara: CreateRegistrationVersion sms-suara: CreateVerifiedDestinationNumber sms-suara: DeleteConfigurationSet sms-suara: DeleteConfigurationSetEventDestination sms-suara: DeleteDefaultMessageType sms-suara: DeleteDefaultSenderId sms-suara: DeleteEventDestination sms-suara: DeleteKeyword sms-suara: DeleteOptedOutNumber sms-suara: DeleteOptOutList sms-suara: DeletePool sms-suara: DeleteRegistration sms-suara: DeleteRegistrationAttachment

Awalan layanan	Tindakan
	sms-suara: DeleteTextMessageSpendLimitOverride
	sms-suara: DeleteVerifiedDestinationNumber
	sms-suara: DeleteVoiceMessageSpendLimitOverride
	sms-suara: DescribeAccountAttributes
	sms-suara: DescribeAccountLimits
	sms-suara: DescribeConfigurationSets
	sms-suara: DescribeKeywords
	sms-suara: DescribeOptedOutNumbers
	sms-suara: DescribeOptOutLists
	sms-suara: DescribePhoneNumbers
	sms-suara: DescribePools
	sms-suara: DescribeRegistrationAttachments
	sms-suara: DescribeRegistrationFieldDefinitions
	sms-suara: DescribeRegistrationFieldValues
	sms-suara: DescribeRegistrations
	sms-suara: DescribeRegistrationSectionDefinitions
	sms-suara: DescribeRegistrationTypeDefinitions
	sms-suara: DescribeRegistrationVersions
	sms-suara: DescribeSenderIds
	sms-suara: DescribeSpendLimits
	sms-suara: DescribeVerifiedDestinationNumbers

Awalan layanan	Tindakan
	sms-suara: DisassociateOriginationIdentity
	sms-suara: DiscardRegistrationVersion
	sms-suara: GetConfigurationSetEventDestinations
	sms-suara: ListConfigurationSets
	sms-suara: ListPoolOriginationIdentities
	sms-suara: ListRegistrationAssociations
	sms-suara: PutKeyword
	sms-suara: PutOptedOutNumber
	sms-suara: ReleasePhoneNumber
	sms-suara: ReleaseSenderId
	sms-suara: RequestPhoneNumber
	sms-suara: RequestSenderId
	sms-suara: SendDestinationNumberVerificationCode
	sms-suara: SetDefaultMessageType
	sms-suara: SetDefaultSenderId
	sms-suara: SetTextMessageSpendLimitOverride
	sms-suara: SetVoiceMessageSpendLimitOverride
	sms-suara: SubmitRegistrationVersion
	sms-suara: UpdateConfigurationSetEventDestination
	sms-suara: UpdateEventDestination
	sms-suara: UpdatePhoneNumber

Awalan layanan	Tindakan
	sms-suara: UpdatePool 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
	<p>bola salju: ListPickupLocations</p> <p>bola salju: ListServiceVersions</p> <p>bola salju: UpdateCluster</p> <p>bola salju: UpdateJob</p> <p>bola salju: UpdateJobShipmentState</p> <p>bola salju: UpdateLongTermPricing</p>
sqs	<p>persegi: AddPermission</p> <p>persegi: CancelMessageMoveTask</p> <p>persegi: CreateQueue</p> <p>persegi: DeleteQueue</p> <p>persegi: PurgeQueue</p> <p>persegi: RemovePermission</p> <p>persegi: SetQueueAttributes</p>

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: DeleteDocument ssm: DeleteInventory ssm: DeleteMaintenanceWindow ssm: DeleteOpsItem ssm: DeleteOpsMetadata ssm: DeleteParameter ssm: DeleteParameters

Awalan layanan	Tindakan
	ssm: DeletePatchBaseline
	ssm: DeleteResourceDataSync
	ssm: DeleteResourcePolicy
	ssm: DeregisterManagedInstance
	ssm: DeregisterPatchBaselineForPatchGroup
	ssm: DeregisterTargetFromMaintenanceWindow
	ssm: DeregisterTaskFromMaintenanceWindow
	ssm: DescribeActivations
	ssm: DescribeAssociation
	ssm: DescribeAssociationExecutions
	ssm: DescribeAssociationExecutionTargets
	ssm: DescribeAutomationExecutions
	ssm: DescribeAutomationStepExecutions
	ssm: DescribeAvailablePatches
	ssm: DescribeDocument
	ssm: DescribeDocumentParameters
	ssm: DescribeDocumentPermission
	ssm: DescribeEffectiveInstanceAssociations
	ssm: DescribeEffectivePatchesForPatchBaseline
	ssm: DescribeInstanceAssociationsStatus
	ssm: DescribeInstanceInformation

Awalan layanan	Tindakan
	ssm: DescribeInstancePatches
	ssm: DescribeInstancePatchStates
	ssm: DescribeInstancePatchStatesForPatchGroup
	ssm: DescribeInstanceProperties
	ssm: DescribeInventoryDeletions
	ssm: DescribeMaintenanceWindowExecutions
	ssm: DescribeMaintenanceWindowExecutionTaskInvocations
	ssm: DescribeMaintenanceWindowExecutionTasks
	ssm: DescribeMaintenanceWindows
	ssm: DescribeMaintenanceWindowSchedule
	ssm: DescribeMaintenanceWindowsForTarget
	ssm: DescribeMaintenanceWindowTargets
	ssm: DescribeMaintenanceWindowTasks
	ssm: DescribeOpsItems
	ssm: DescribeParameters
	ssm: DescribePatchBaselines
	ssm: DescribePatchGroups
	ssm: DescribePatchGroupState
	ssm: DescribePatchProperties
	ssm: DescribeSessions
	ssm: DisassociateOpsItemRelatedItem

Awalan layanan	Tindakan
	ssm: GetAutomationExecution
	ssm: GetCalendarState
	ssm: GetCommandInvocation
	ssm: GetConnectionStatus
	ssm: GetDefaultPatchBaseline
	ssm: GetDeployablePatchSnapshotForInstance
	ssm: GetDocument
	ssm: GetInventory
	ssm: GetInventorySchema
	ssm: GetMaintenanceWindow
	ssm: GetMaintenanceWindowExecution
	ssm: GetMaintenanceWindowExecutionTask
	ssm: GetMaintenanceWindowExecutionTaskInvocation
	ssm: GetMaintenanceWindowTask
	ssm: GetOpsItem
	ssm: GetOpsMetadata
	ssm: GetOpsSummary
	ssm: GetParameter
	ssm: GetParameterHistory
	ssm: GetParameters
	ssm: GetParametersByPath

Awalan layanan	Tindakan
	ssm: GetPatchBaseline
	ssm: GetPatchBaselineForPatchGroup
	ssm: GetResourcePolicies
	ssm: GetServiceSetting
	ssm: LabelParameterVersion
	ssm: ListAssociations
	ssm: ListAssociationVersions
	ssm: ListCommandInvocations
	ssm: ListCommands
	ssm: ListComplianceItems
	ssm: ListComplianceSummaries
	ssm: ListDocumentMetadataHistory
	ssm: ListDocuments
	ssm: ListDocumentVersions
	ssm: ListInstanceAssociations
	ssm: ListInventoryEntries
	ssm: ListOpsItemEvents
	ssm: ListOpsItemRelatedItems
	ssm: ListOpsMetadata
	ssm: ListResourceComplianceSummaries
	ssm: ListResourceDataSync

Awalan layanan	Tindakan
	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
	ssm: StartAssociationsOnce
	ssm: StartAutomationExecution
	ssm: StartChangeRequestExecution
	ssm: StartSession
	ssm: StopAutomationExecution
	ssm: TerminateSession
	ssm: UnlabelParameterVersion

Awalan layanan	Tindakan
	ssm: UpdateAssociation
	ssm: UpdateAssociationStatus
	ssm: UpdateDocument
	ssm: UpdateDocumentDefaultVersion
	ssm: UpdateDocumentMetadata
	ssm: UpdateInstanceInformation
	ssm: UpdateMaintenanceWindow
	ssm: UpdateMaintenanceWindowTarget
	ssm: UpdateMaintenanceWindowTask
	ssm: UpdateManagedInstanceRole
	ssm: UpdateOpsItem
	ssm: UpdateOpsMetadata
	ssm: UpdatePatchBaseline
	ssm: UpdateResourceDataSync
	ssm: UpdateServiceSetting

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: GetDatabase ssm-sap: GetOperation ssm-sap: GetResourcePermission ssm-sap: ListApplications ssm-sap: ListComponents ssm-sap: ListDatabases ssm-sap: ListOperations ssm-sap: PutResourcePermission ssm-sap: RegisterApplication ssm-sap: RestoreDatabase ssm-sap: StartApplicationRefresh ssm-sap: UpdateApplicationSettings SSM-SAP: Updatehana BackupSettings

Awalan layanan	Tindakan
negara	negara bagian: CreateActivity negara bagian: CreateStateMachine negara bagian: CreateStateMachineAlias negara bagian: DeleteActivity negara bagian: DeleteStateMachine negara bagian: DeleteStateMachineAlias negara bagian: DeleteStateMachineVersion negara bagian: DescribeActivity negara bagian: DescribeExecution negara bagian: DescribeMapRun negara bagian: DescribeStateMachine negara bagian: DescribeStateMachineAlias negara bagian: DescribeStateMachineForExecution negara bagian: GetExecutionHistory negara bagian: ListActivities negara bagian: ListExecutions negara bagian: ListMapRuns negara bagian: ListStateMachineAliases negara bagian: ListStateMachines negara bagian: ListStateMachineVersions negara bagian: 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
sts	sts: AssumeRole sts: AssumeRoleWith SALL sts: AssumeRoleWithWebIdentity sts: DecodeAuthorizationMessage sts: GetAccessKeyInfo sts: GetCallerIdentity sts: GetFederationToken sts: GetSessionToken

Awalan layanan	Tindakan
swf	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: 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: ListAdapters teks: ListAdapterVersions teks: StartDocumentAnalysis teks: StartDocumentTextDetection teks: StartExpenseAnalysis teks: StartLendingAnalysis

Awalan layanan	Tindakan
	teks: UpdateAdapter
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: 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: 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: CreateWorkflow transfer: DeleteAccess transfer: DeleteAgreement transfer: DeleteCertificate transfer: DeleteConnector transfer: DeleteHostKey transfer: DeleteProfile transfer: DeleteServer transfer: DeleteSshPublicKey transfer: DeleteUser transfer: DeleteWorkflow transfer: DescribeAccess transfer: DescribeAgreement transfer: DescribeCertificate transfer: DescribeConnector

Awalan layanan	Tindakan
	transfer: DescribeExecution
	transfer: DescribeHostKey
	transfer: DescribeProfile
	transfer: DescribeSecurityPolicy
	transfer: DescribeServer
	transfer: DescribeUser
	transfer: DescribeWorkflow
	transfer: ImportCertificate
	transfer: ImportHostKey
	transfer: ImportSshPublicKey
	transfer: ListAccesses
	transfer: ListCertificates
	transfer: ListConnectors
	transfer: ListExecutions
	transfer: ListHostKeys
	transfer: ListProfiles
	transfer: ListSecurityPolicies
	transfer: ListServers
	transfer: ListUsers
	transfer: ListWorkflows
	transfer: SendWorkflowStepState

Awalan layanan	Tindakan
	transfer: StartFileTransfer
	transfer: StartServer
	transfer: StopServer
	transfer: TestConnection
	transfer: TestIdentityProvider
	transfer: UpdateAccess
	transfer: UpdateAgreement
	transfer: UpdateCertificate
	transfer: UpdateConnector
	transfer: UpdateHostKey
	transfer: UpdateProfile
	transfer: UpdateServer
	transfer: UpdateUser

Awalan layanan	Tindakan
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: CreateRule vpc-kisi: CreateService vpc-kisi: CreateServiceNetwork vpc-kisi: CreateServiceNetworkServiceAssociation vpc-kisi: CreateServiceNetworkVpcAssociation vpc-kisi: CreateTargetGroup vpc-kisi: DeleteAccessLogSubscription vpc-kisi: DeleteAuthPolicy vpc-kisi: DeleteListener vpc-kisi: DeleteResourcePolicy vpc-kisi: DeleteRule vpc-kisi: DeleteService vpc-kisi: DeleteServiceNetwork vpc-kisi: DeleteServiceNetworkServiceAssociation vpc-kisi: DeleteServiceNetworkVpcAssociation vpc-kisi: DeleteTargetGroup vpc-kisi: DeregisterTargets vpc-kisi: GetAccessLogSubscription vpc-kisi: GetAuthPolicy

Awalan layanan	Tindakan
	vpc-kisi: GetListener
	vpc-kisi: GetResourcePolicy
	vpc-kisi: GetRule
	vpc-kisi: GetService
	vpc-kisi: GetServiceNetwork
	vpc-kisi: GetServiceNetworkServiceAssociation
	vpc-kisi: GetServiceNetworkVpcAssociation
	vpc-kisi: GetTargetGroup
	vpc-kisi: ListAccessLogSubscriptions
	vpc-kisi: ListListeners
	vpc-kisi: ListRules
	vpc-kisi: ListServiceNetworks
	vpc-kisi: ListServiceNetworkServiceAssociations
	vpc-kisi: ListServiceNetworkVpcAssociations
	vpc-kisi: ListServices
	vpc-kisi: ListTargetGroups
	vpc-kisi: ListTargets
	vpc-kisi: PutAuthPolicy
	vpc-kisi: PutResourcePolicy
	vpc-kisi: RegisterTargets
	vpc-kisi: UpdateAccessLogSubscription

Awalan layanan	Tindakan
	vpc-kisi: UpdateListener 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: apiKey GetDecrypted

Awalan layanan	Tindakan
	WAFV2: GetIpSet
	wafv2: GetLoggingConfiguration
	wafv2: GetManagedRuleSet
	wafv2: GetMobileSdkRelease
	wafv2: GetPermissionPolicy
	wafv2: GetRateBasedStatementManagedKeys
	wafv2: GetRegexPatternSet
	wafv2: GetRuleGroup
	wafv2: GetSampledRequests
	wafv2: ACL GetWeb ForResource
	WAFv2:ListApiKeys
	wafv2: ListAvailableManagedRuleGroups
	wafv2: ListAvailableManagedRuleGroupVersions
	WAFv2:ListipSets
	wafv2: ListLoggingConfigurations
	wafv2: ListManagedRuleSets
	wafv2: ListMobileSdkReleases
	wafv2: ListRegexPatternSets
	wafv2: ACL ListResourcesForWeb
	wafv2: ListRuleGroups
	wafv2: ACL ListWeb

Awalan layanan	Tindakan
	<p>wafv2: PutLoggingConfiguration</p> <p>wafv2: PutManagedRuleSetVersions</p> <p>wafv2: PutPermissionPolicy</p> <p>WAFv2: UpdateIPSet</p> <p>wafv2: UpdateManagedRuleSetVersionExpiryDate</p> <p>wafv2: UpdateRegexPatternSet</p> <p>wafv2: UpdateRuleGroup</p> <p>wafv2: ACL UpdateWeb</p>

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: 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: ListLenses
	wellarchitected: ListLensReviewImprovements
	wellarchitected: ListLensReviews
	wellarchitected: ListLensShares

Awalan layanan	Tindakan
	<p>wellarchitected: ListMilestones</p> <p>wellarchitected: ListNotifications</p> <p>wellarchitected: ListProfileNotifications</p> <p>wellarchitected: ListProfiles</p> <p>wellarchitected: ListProfileShares</p> <p>wellarchitected: ListReviewTemplateAnswers</p> <p>wellarchitected: ListReviewTemplates</p> <p>wellarchitected: ListShareInvitations</p> <p>wellarchitected: ListTemplateShares</p> <p>wellarchitected: ListWorkloads</p> <p>wellarchitected: ListWorkloadShares</p> <p>wellarchitected: UpdateAnswer</p> <p>wellarchitected: UpdateGlobalSettings</p> <p>wellarchitected: UpdateLensReview</p> <p>wellarchitected: UpdateProfile</p> <p>wellarchitected: UpdateReviewTemplate</p> <p>wellarchitected: UpdateReviewTemplateLensReview</p> <p>wellarchitected: UpdateShareInvitation</p> <p>wellarchitected: UpdateWorkload</p> <p>wellarchitected: UpdateWorkloadShare</p> <p>wellarchitected: UpgradeLensReview</p>

Awalan layanan	Tindakan
	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: GetContentSummary kebijaksanaan: GetImportJob kebijaksanaan: GetKnowledgeBase kebijaksanaan: GetRecommendations kebijaksanaan: GetSession kebijaksanaan: ListAssistantAssociations

Awalan layanan	Tindakan
	kebijaksanaan: ListAssistants
	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

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
	<p>tautan kerja: RestoreDomainAccess</p> <p>tautan kerja: RevokeDomainAccess</p> <p>tautan kerja: SignOutUser</p> <p>tautan kerja: UpdateAuditStreamConfiguration</p> <p>tautan kerja: UpdateCompanyNetworkConfiguration</p> <p>tautan kerja: UpdateDevicePolicyConfiguration</p> <p>tautan kerja: UpdateDomainMetadata</p> <p>tautan kerja: UpdateFleetMetadata</p> <p>tautan kerja: UpdateIdentityProviderConfiguration</p>

Awalan layanan	Tindakan
ruang kerja	ruang kerja: AssociateConnectionAlias ruang kerja: AssociateIpgroups ruang kerja: AssociateWorkspaceApplication ruang kerja: CopyWorkspacelImage ruang kerja: CreateConnectClientAddIn ruang kerja: CreateConnectionAlias ruang kerja: CreateIpgroup ruang kerja: CreateStandbyWorkspaces ruang kerja: CreateUpdatedWorkspacelImage ruang kerja: CreateWorkspaceBundle ruang kerja: CreateWorkspacelImage ruang kerja: CreateWorkspaces ruang kerja: DeleteClientBranding ruang kerja: DeleteConnectClientAddIn ruang kerja: DeleteConnectionAlias ruang kerja: DeleteIpgroup ruang kerja: DeleteWorkspaceBundle ruang kerja: DeleteWorkspacelImage ruang kerja: DeployWorkspaceApplications ruang kerja: DeregisterWorkspaceDirectory ruang kerja: DescribeAccount

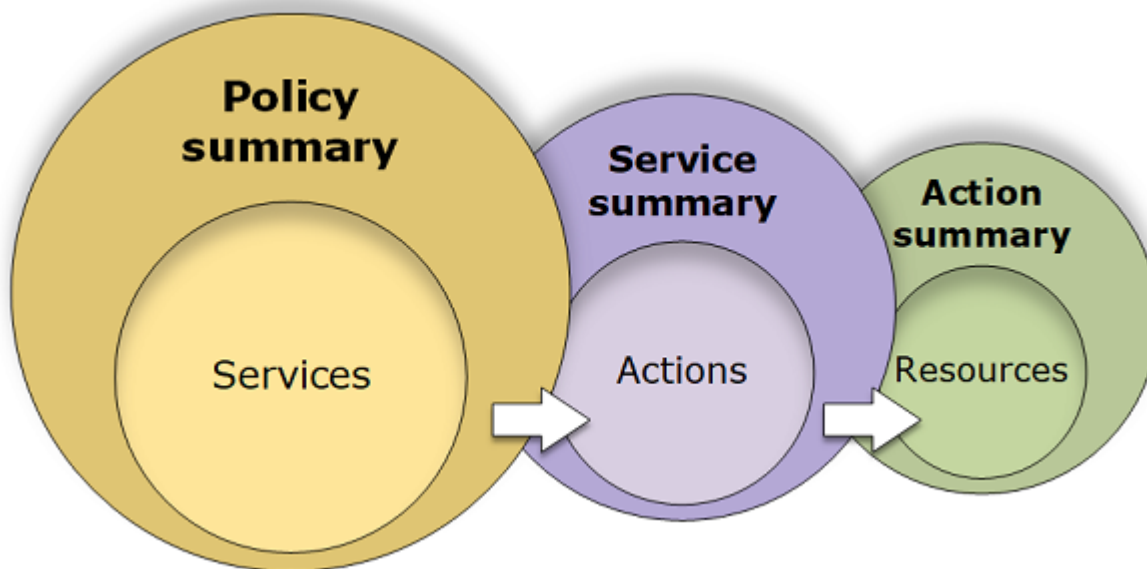
Awalan layanan	Tindakan
	ruang kerja: DescribeAccountModifications
	ruang kerja: DescribeApplicationAssociations
	ruang kerja: DescribeApplications
	ruang kerja: DescribeBundleAssociations
	ruang kerja: DescribeClientBranding
	ruang kerja: DescribeClientProperties
	ruang kerja: DescribeConnectClientAddIns
	ruang kerja: DescribeConnectionAliases
	ruang kerja: DescribeConnectionAliasPermissions
	ruang kerja: DescribeImageAssociations
	ruang kerja: DescribeIpGroups
	ruang kerja: DescribeWorkspaceAssociations
	ruang kerja: DescribeWorkspaceBundles
	ruang kerja: DescribeWorkspaceDirectories
	ruang kerja: DescribeWorkspacelImagePermissions
	ruang kerja: DescribeWorkspaces
	ruang kerja: DescribeWorkspacesConnectionStatus
	ruang kerja: DescribeWorkspaceSnapshots
	ruang kerja: DisassociateConnectionAlias
	ruang kerja: DisassociateIpGroups
	ruang kerja: DisassociateWorkspaceApplication

Awalan layanan	Tindakan
	ruang kerja: ImportClientBranding
	ruang kerja: ImportWorkpaceImage
	ruang kerja: ListAvailableManagementCidrRanges
	ruang kerja: MigrateWorkspace
	ruang kerja: ModifyAccount
	ruang kerja: ModifyCertificateBasedAuthProperties
	ruang kerja: ModifyClientProperties
	ruang kerja: ModifySamlProperties
	ruang kerja: ModifySelfservicePermissions
	ruang kerja: ModifyWorkspaceAccessProperties
	ruang kerja: ModifyWorkspaceCreationProperties
	ruang kerja: ModifyWorkspaceProperties
	ruang kerja: ModifyWorkspaceState
	ruang kerja: RebootWorkspaces
	ruang kerja: RebuildWorkspaces
	ruang kerja: RegisterWorkspaceDirectory
	ruang kerja: RestoreWorkspace
	ruang kerja: StartWorkspaces
	ruang kerja: StopWorkspaces
	ruang kerja: TerminateWorkspaces
	ruang kerja: UpdateConnectClientAddIn

Awalan layanan	Tindakan
	ruang kerja: UpdateConnectionAliasPermission ruang kerja: UpdateWorkspaceBundle ruang kerja: UpdateWorkspaceImagePermission
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

Memahami izin yang diberikan oleh 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 terkelola mencakup kebijakan terkelola AWS, kebijakan fungsi pekerjaan terkelola AWS, dan kebijakan terkelola 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.

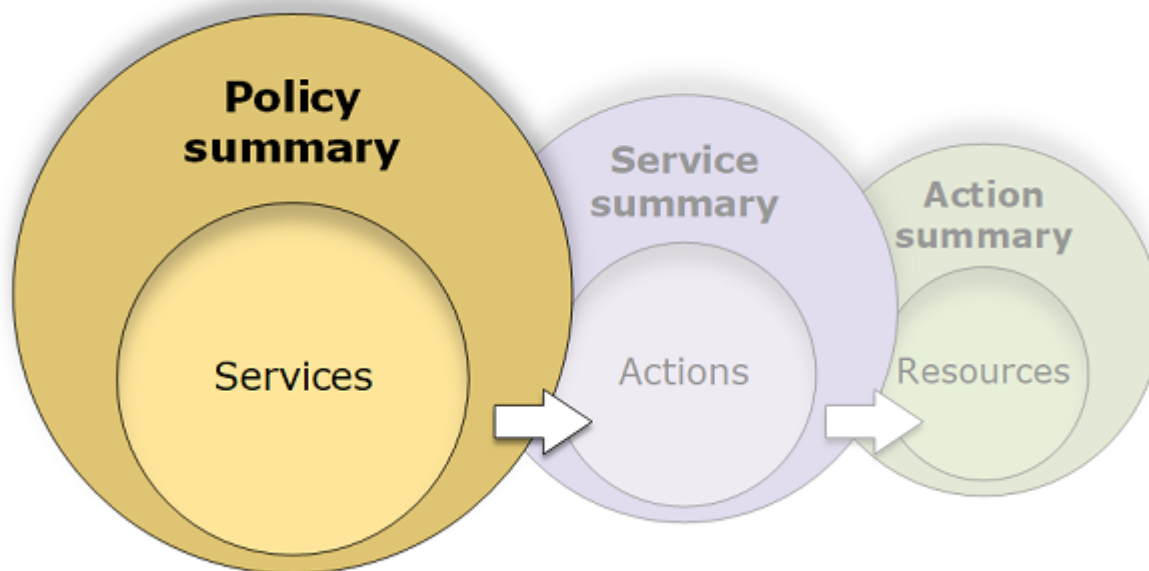
Topik

- [Ringkasan kebijakan \(daftar layanan\)](#)
- [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 terkategori, Penolakan secara eksplisit, dan Izinkan. Jika kebijakan mencakup layanan yang tidak dikenali oleh IAM, layanan tersebut termasuk dalam Layanan yang tidak terkategori di tabel. Jika IAM mengenali layanan, maka ia termasuk dalam bagian tabel Penolakan secara eksplisit atau Izinkan tergantung pada efek kebijakan (Deny atau Allow).

Melihat ringkasan kebijakan

Anda dapat melihat ringkasan untuk setiap kebijakan yang dilampirkan ke pengguna dengan memilih nama kebijakan pada tab Izin di halaman detail pengguna. Anda dapat melihat ringkasan untuk kebijakan apa pun yang dilampirkan ke peran dengan memilih nama kebijakan di tab Izin di halaman detail peran. Anda dapat melihat ringkasan kebijakan untuk kebijakan dikelola di Kebijakan yang baru. Jika kebijakan Anda tidak menyertakan ringkasan kebijakan, lihat [Ringkasan kebijakan hilang](#) untuk mempelajari alasannya.

Untuk melihat ringkasan kebijakan dari halaman Kebijakan

1. Masuk ke AWS Management Console 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.

Untuk melihat ringkasan kebijakan yang terlampir pada pengguna

1. Masuk ke AWS Management Console 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 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.

Untuk melihat ringkasan kebijakan yang terlampir pada peran

1. Masuk ke AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Di panel navigasi, silakan pilih Peran.
3. Dalam daftar peran, pilih nama peran yang kebijakannya ingin Anda lihat.
4. Di Ringkasan untuk peran, lihat Izin untuk melihat daftar kebijakan yang terlampir pada peran tersebut.
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 “Menyunting Kebijakan IAM”](#). Kebijakan terkelola AWS tidak dapat diedit.

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.

Untuk mengedit kebijakan ringkasan kebijakan Anda dengan 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.

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.

Policy details

Type Customer managed	Creation time September 13, 2022, 16:37 (UTC-05:00)	Edited time September 13, 2022, 16:40 (UTC-05:00)	ARN arn:aws:iam:::policy/SummaryAllElements
--------------------------	--	--	--

1

Permissions Entitles attached Tags Policy versions Access Advisor

2

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. For details, choose [Show remaining](#). [Learn more](#)

3

Permissions defined in this policy [info](#)

Permissions defined in this policy document specify which actions are allowed or denied. To define permissions for an IAM identity (user, user group, or role), attach a policy to it.

4

Search

5

Explicit deny (1 of 338 services)

Service	Access level	Resource	Request condition
S3	Limited: List, Permissions management, Read, Write, Tagging	Multiple	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
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:

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”](#).

- Gunakan tombol Ringkasan dan JSON untuk beralih antara ringkasan kebijakan dan dokumen kebijakan JSON.
- Gunakan kotak Pencarian untuk mengurangi daftar layanan dan menemukan layanan tertentu.
- 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 <input type="checkbox"/> 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:

- 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.
- 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.
- Tingkat akses — Kolom ini memberi tahu apakah tindakan di setiap tingkat akses (List,Read,Write,Permission Management,, danTagging) memiliki Full atau Limited izin

yang ditentukan dalam kebijakan. Untuk detail dan contoh tambahan dari ringkasan tingkat akses, lihat [Memahami 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 layanan AWS memutuskan apakah permintaan tertentu diizinkan atau ditolak, lihat [Logika 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) <input type="checkbox"/> 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-ac1`. 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.

```
{
  "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:DeleteObject",
        "s3:PutObject",
        "s3:PutObjectAcl"
    ],
    "Resource": [
        "arn:aws:s3:::developer_bucket",
        "arn:aws:s3:::developer_bucket/*",
        "arn:aws:autoscaling:us-east-2:123456789012:autoscalgrp"
    ],
    "Condition": {
        "StringEquals": {
            "s3:x-amz-acl": [
                "public-read"
            ],
            "s3:prefix": [
```

```

    "custom",
    "other"
  ]
}
}
}
]
}

```


Memahami 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 [Memahami izin yang diberikan oleh 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.
CloudWatch	Penuh:Daftar	Akses ke semua CloudWatch tindakan di tingkat <code>List</code> akses, tetapi tidak ada akses ke tindakan dengan <code>Read</code> , <code>Write</code> , atau klasifikasi tingkat <code>Permissions management</code> akses.
Data Pipeline	Terbatas: Daftar, Baca	Akses ke setidaknya satu tetapi tidak semua AWS Data Pipeline tindakan di <code>List</code> dan tingkat <code>Read</code> akses, tetapi tidak <code>Write</code> atau <code>Permissions management</code> tindakan.

Layanan	Tingkat akses	Kebijakan memungkinkan hal-hal berikut
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 .
CodeDeploy	(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” .

Sebagaimana [yang disebutkan sebelumnya](#), Akses penuh menunjukkan bahwa kebijakan tersebut memberikan 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 akses `List`, `Read`, `Tagging` atau `Write 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 Tulis. `Write` tindakan mungkin juga memungkinkan memodifikasi tag sumber daya. Namun, tindakan yang hanya memungkinkan perubahan pada tanda memiliki tingkat akses `Tagging`.
- **Manajemen izin** Izin untuk memberikan atau mengubah izin sumber daya dalam layanan. Misalnya, sebagian besar IAM dan AWS Organizations tindakan, serta tindakan seperti tindakan Amazon S3 `PutBucketPolicy` `DeleteBucketPolicy` dan memiliki tingkat akses manajemen Izin.

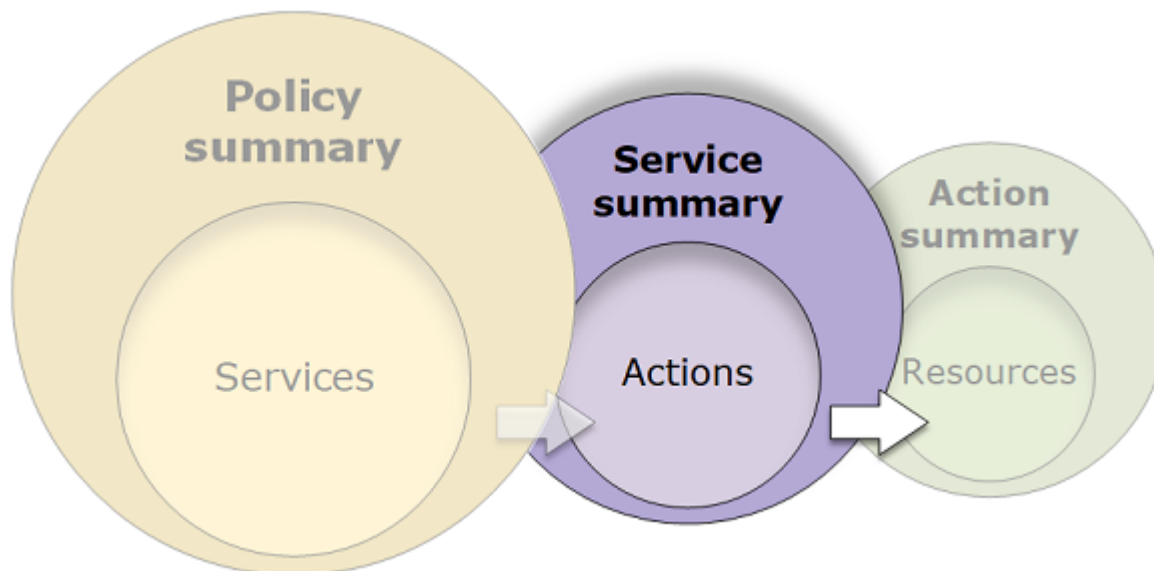
i Kiat

Untuk meningkatkan keamanan Anda Akun AWS, batasi atau pantau secara teratur kebijakan yang mencakup 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 untuk setiap tindakan dalam layanan yang ditetapkan, lihat [Tindakan, Sumber Daya, dan Kunci Syarat untuk Layanan AWS](#).

Melihat ringkasan layanan

Anda dapat melihat ringkasan layanan untuk kebijakan terkelola di halaman Kebijakan.

Untuk melihat ringkasan layanan untuk kebijakan yang dikelola

1. Masuk ke AWS Management Console 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.

Untuk melihat ringkasan layanan untuk kebijakan yang terlampir pada pengguna

1. Masuk ke AWS Management Console 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 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.

Untuk melihat ringkasan layanan untuk kebijakan yang terlampir pada peran

1. Masuk ke AWS Management Console 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 Ringkasan untuk peran, lihat Izin untuk melihat daftar kebijakan yang terlampir pada peran tersebut.
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.

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 didefinisikan dari total 52 tindakan Baca yang tersedia untuk layanan.

Permissions

Entities attached

Tags

Policy versions

Access Advisor

i 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. For details, choose **Show remaining**. [Learn more](#)

Permissions defined in this policy [Info](#)

Permissions defined in this policy document specify which actions are allowed or denied. To define permissions for an IAM identity (user, user group, or role), attach a policy to it.

Edit

Summary

JSON

< Services Actions in S3 (82 of 128)

Read (35 of 52)

 Show remaining 46 actions

Action

Resource

Request condition

DescribeJob (No access)

! This action does not have an applicable resource.

None

DescribeMultiRegionAccessPointOperation (No access)

! This action does not have an applicable resource.

None

GetAccelerateConfiguration

BucketName | string like | customer

None

GetAccessPoint (No access)

! This action does not have an applicable resource.

None

GetAccessPointConfigurationForObjectLambda (No access)

! This action does not have an applicable resource.

None

GetAccessPointForObjectLambda (No access)

! This action does not have an applicable resource.

None

GetAccessPointPolicy (No access)

! This action does not have an applicable resource.

None

GetAccessPointPolicyForObjectLambda (No access)

! This action does not have an applicable resource.

None

GetAccessPointPolicyStatus (No access)

! This action does not have an applicable resource.

None

GetAccessPointPolicyStatusForObjectLambda (No access)

! This action does not have an applicable resource.

None

GetAccountPublicAccessBlock (No access)

! This action does not have an applicable resource.

None

GetAnalyticsConfiguration

BucketName | string like | customer

None

GetBucketAcl

BucketName | string like | customer

None


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 tersebut. 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 belakang Layanan muncul nama layanan (dalam hal ini S3). Ringkasan layanan untuk layanan ini mencakup daftar tindakan yang diizinkan atau ditolak yang didefinisikan 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. Tindakan - 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 ditautkan 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 tersebut. Bagian-bagiannya adalah Daftar, Baca, Tulis, Manajemen Izin, dan Penandaan. Hitungan menunjukkan jumlah tindakan yang diakui yang memberikan izin dalam setiap tingkat akses. Totalnya adalah jumlah tindakan yang diketahui untuk layanan tersebut. Dalam contoh ini, 35 tindakan memberikan izin dari 52 total tindakan Amazon S3 Read yang diketahui. Untuk melihat klasifikasi tingkat akses untuk setiap tindakan dalam layanan yang ditetapkan, lihat [Tindakan, Sumber Daya, dan Kunci Ketentuan untuk Layanan AWS](#).
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. Mengaktifkan 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 Amazon `developer_bucket` 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`.

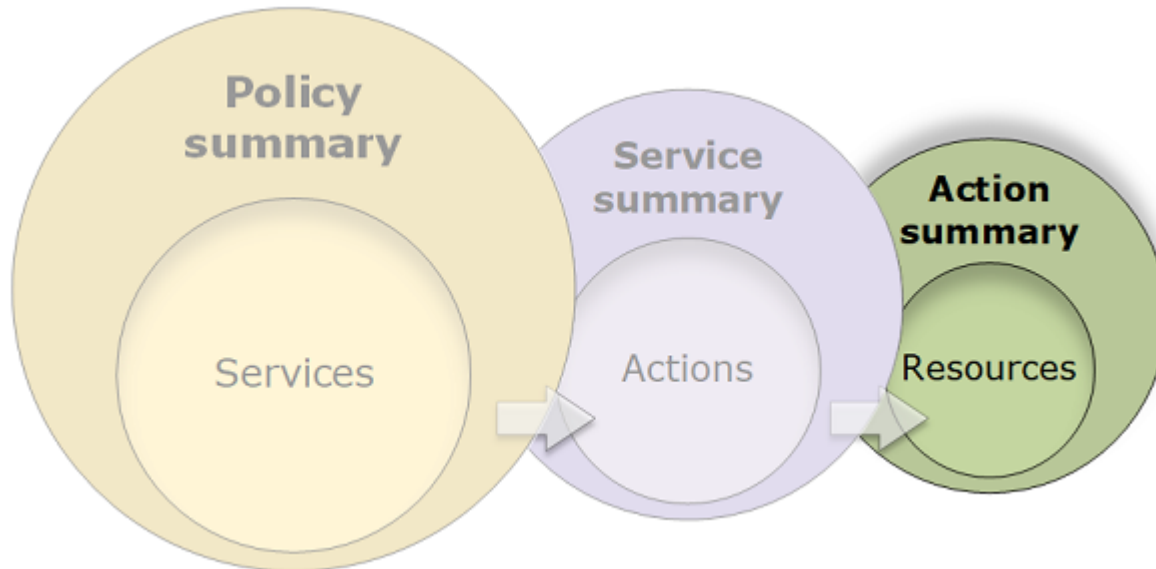
 Note

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.

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.

Melihat ringkasan tindakan

Anda dapat melihat ringkasan tindakan untuk kebijakan terkelola, kebijakan apa pun yang dilampirkan ke pengguna, dan kebijakan apa pun yang dilampirkan ke peran di halaman Kebijakan.

Untuk melihat ringkasan tindakan untuk kebijakan yang dikelola

1. Masuk ke AWS Management Console 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.

Untuk melihat ringkasan tindakan untuk kebijakan yang terlampir pada pengguna

1. Masuk ke AWS Management Console 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 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.

Untuk melihat ringkasan tindakan untuk kebijakan yang terlampir pada peran

1. Masuk ke AWS Management Console 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 Ringkasan untuk peran, lihat Izin untuk melihat daftar kebijakan yang terlampir pada peran tersebut.

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.

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.

Permissions defined in this policy [Info](#)

Permissions defined in this policy document specify which actions are allowed or denied. To define permissions for an [IAM](#) identity (user, user group, or role), attach a policy to it.

[Edit](#) [Summary](#) [JSON](#)

[< Actions](#) PutObject action in S3

Resource	Region	Account	Request condition
BucketName string like customer, ObjectPath string like All	All regions	All accounts	s3:x-amz-acl = public-read

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 in service` (dalam hal ini PutObjecttindakan 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.

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.

```
{
  "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:

i 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. For details, choose **Show remaining**. [Learn more](#)

Permissions defined in this policy [Info](#)

Permissions defined in this policy document specify which actions are allowed or denied. To define permissions for an IAM identity (user, user group, or role), attach a policy to it

[Edit](#) [Summary](#) [JSON](#)

Q Search

Explicit deny (1 of 371 services)

Service	Access level	Resource	Request condition
S3	Limited: List, Permissions management, Read, Write, Tagging	Multiple	None

Allow (1 of 371 services)

Show remaining 369 services

Service	Access level	Resource	Request condition
S3	Full access	All resources	None

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 Tindakan:

< Actions GetObject action in S3

Resource	Region	Account	Request condition
BucketName string like customer, ObjectPath string like All	-	All accounts	None

Kebijakan 2: DynamoDbRowCognito ID

Kebijakan ini menyediakan akses tingkat baris ke Amazon DynamoDB berdasarkan ID Amazon Cognito pengguna.

```
{
```

```

"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}"
        ]
      }
    }
  }
]
}

```

DynamoDbRowCognitoRingkasan Kebijakan ID:

Allow (1 of 370 services)		<input type="checkbox"/> Show remaining 369 services	
Service	Access level	Resource	Request condition
DynamoDB	Limited: Read, Write	region string like us-west-1, TableName string like myDynamoTable	dynamodb:LeadingKeys = \${cognito-identity.amazonaws.com:sub}

DynamoDbRowCognitoID DynamoDB (Izinkan) Ringkasan Layanan:

< 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 Tindakan:

< 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.

```
{
  "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"]
    }}
  }
]
}

```

MultipleResourceCondition Ringkasan 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 Tindakan:

< 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.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:GetConsoleScreenshot"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:ListBucket"
      ],
      "Resource": [
        "arn:aws:s3:::developer"
      ]
    }
  ]
}
```

EC2_Troubleshoot Ringkasan Kebijakan:

Allow (2 of 370 services) Show remaining 368 services			
Service	Access level	Resource	Request condition
EC2	Limited: Read	All resources	None
S3	Limited: List	BucketName string like developer	None

EC2_Troubleshoot S3 (Izinkan) Ringkasan Layanan:

Action	Resource	Request condition
ListBucket	BucketName string like developer	None

ListBucket (Daftar) Ringkasan Tindakan:

Resource	Region	Account	Request condition
BucketName string like developer	-	All accounts	None

Kebijakan 5: CodeBuild _ CodeCommit _ CodeDeploy

Kebijakan ini menyediakan akses ke sumber daya tertentu CodeBuild CodeCommit, dan CodeDeploy sumber daya. 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.

```
{
  "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)			<input type="checkbox"/> 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 Tindakan:

< 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 kredensial, 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

Di seluruh AWS dokumentasi, jika kami merujuk ke sebuah kebijakan IAM tanpa menyebutkan kategori tertentu, yang kami maksudkan adalah kebijakan pengelolaan pelanggan berbasis identitas. 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 terkait: [CreateUser](#). Untuk mengizinkan pengguna IAM membuat pengguna IAM lain, Anda dapat melampirkan kebijakan IAM seperti yang berikut kepada pengguna tersebut:

```
{
  "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:

```
{
  "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-variabel 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 future yang memiliki `AccessKey` dalam nama, menggunakan `iam:*AccessKey` untuk `Action` elemen juga akan memberikan izin pengguna untuk tindakan baru.) Contoh berikut menunjukkan kebijakan yang mengizinkan pengguna melakukan semua tindakan yang berkaitan dengan access key mereka sendiri (ganti `account-id` dengan Akun AWS ID Anda):

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:*AccessKey*",
    "Resource": "arn:aws:iam::account-id: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 AWS Management Console

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 prinsipal. 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 memberi pengguna izin untuk mengatur grup, pengguna, peran, kebijakan, dan kredensial dengan AWS Management Console, Anda perlu 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](#).

Memberikan izin di seluruh akun AWS

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 [Menyediakan akses ke pengguna IAM di pengguna lain Akun AWS yang Anda miliki](#).

Note

Beberapa layanan mendukung kebijakan berbasis sumber daya sebagaimana dijelaskan dalam [Kebijakan berbasis identitas 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 antrian) 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 layanan AWS mengakses layanan AWS lainnya. Misalnya, beberapa AWS layanan — 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.
- Dalam AWS Data Pipeline, peran IAM menentukan apa yang dapat dilakukan saluran pipa; pengguna memerlukan izin untuk mengambil peran tersebut. (Untuk detailnya, lihat [Memberikan Izin ke Pipelines dengan IAM](#) di Panduan AWS Data Pipeline Pengembang.)

Untuk detail tentang cara mengonfigurasi izin dengan benar sehingga layanan AWS dapat menyelesaikan tugas yang Anda inginkan, lihat dokumentasi untuk layanan yang Anda panggil. Untuk mempelajari cara membuat peran untuk layanan, lihat [Membuat peran untuk mendelegasikan izin ke layanan AWS](#).

Mengonfigurasi layanan dengan peran IAM untuk bekerja atas nama Anda

Jika Anda ingin mengonfigurasi AWS layanan untuk bekerja atas nama Anda, Anda biasanya memberikan ARN untuk peran IAM yang menentukan apa yang dapat dilakukan layanan tersebut. AWS memeriksa untuk memastikan bahwa Anda memiliki izin untuk memberikan peran ke layanan. Untuk informasi selengkapnya, lihat [Memberi izin kepada pengguna untuk meneruskan peran ke layanan AWS](#).

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 layanan AWS.

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 terhadap 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 lebih lanjut tentang izin yang diperlukan untuk membuat direktori di AWS Directory Service, lihat [Contoh 2: Memungkinkan Pengguna untuk 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.

Sebagai contoh kebijakan yang memungkinkan pengguna melaksanakan tugas dengan AWS layanan lainnya, 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-](#).

Memungkinkan pengguna untuk mengelola keanggotaan grup

Kebijakan berikut memungkinkan pengguna memperbarui keanggotaan kelompok yang dipanggil MarketingGroup. Untuk melihat contoh kebijakan, lihat [IAM: Memungkinkan pengelolaan 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 [ListGroupsWithUser](#)).
- 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](#).

```
{
  "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 AWS Management Console. Pengguna yang melakukan tugas terkait pengguna dari [AWS CLI](#), [AWSSDK](#), atau permintaan API HTTP IAM hanya 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 AWS Management Console:

- `iam:GetAccount*`
- `iam:ListAccount*`

Izinkan pengguna mengatur kebijakan kata sandi akun

Anda dapat memberi beberapa pengguna izin untuk mendapatkan dan memperbarui [kebijakan kata sandi](#) AndaAkun AWS. Untuk melihat contoh kebijakan, lihat [IAM: Mengizinkan pengaturan persyaratan kata sandi secara terprogram dan di konsol](#).

Memungkinkan pengguna membuat dan mengambil laporan kredensial IAM

Anda dapat memberi pengguna izin untuk membuat dan mengunduh laporan yang mencantumkan semua pengguna di akun AndaAkun 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 [Mendapatkan laporan kredensi untuk Anda Akun AWS](#). Untuk melihat contoh kebijakan, lihat [IAM: Buat dan ambil 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

Jika Anda memberikan akses penuh kepada pengguna ke IAM, tidak ada batasan untuk izin yang dapat diberikan pengguna kepadanya atau orang lain. Pengguna dapat membuat entitas IAM baru (pengguna atau peran) dan memberi entitas tersebut akses penuh ke semua sumber daya di AndaAkun AWS. Saat Anda memberi pengguna akses penuh ke IAM, Anda secara efektif memberi mereka akses penuh ke semua sumber daya di AndaAkun 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.

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

```
}
```

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, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Contoh kode

- [Contoh kode untuk IAM menggunakan AWS SDK](#)
 - [Tindakan untuk IAM menggunakan AWS SDK](#)
 - [Gunakan AddClientIdToOpenIdConnectProvider dengan AWS SDK atau alat baris perintah](#)
 - [Gunakan AddRoleToInstanceProfile dengan AWS SDK atau alat baris perintah](#)
 - [Gunakan AddUserToGroup dengan AWS SDK atau alat baris perintah](#)
 - [Gunakan AttachGroupPolicy dengan AWS SDK atau alat baris perintah](#)
 - [Gunakan AttachRolePolicy dengan AWS SDK atau alat baris perintah](#)
 - [Gunakan AttachUserPolicy dengan AWS SDK atau alat baris perintah](#)
 - [Gunakan ChangePassword dengan AWS SDK atau alat baris perintah](#)
 - [Gunakan CreateAccessKey dengan AWS SDK atau alat baris perintah](#)
 - [Gunakan CreateAccountAlias dengan AWS SDK atau alat baris perintah](#)
 - [Gunakan CreateGroup dengan AWS SDK atau alat baris perintah](#)
 - [Gunakan CreateInstanceProfile dengan AWS SDK atau alat baris perintah](#)
 - [Gunakan CreateLoginProfile dengan AWS SDK atau alat baris perintah](#)
 - [Gunakan CreateOpenIdConnectProvider dengan AWS SDK atau alat baris perintah](#)
 - [Gunakan CreatePolicy dengan AWS SDK atau alat baris perintah](#)
 - [Gunakan CreatePolicyVersion dengan AWS SDK atau alat baris perintah](#)
 - [Gunakan CreateRole dengan AWS SDK atau alat baris perintah](#)
 - [Gunakan CreateSAMLProvider dengan AWS SDK atau alat baris perintah](#)
 - [Gunakan CreateServiceLinkedRole dengan AWS SDK atau alat baris perintah](#)
 - [Gunakan CreateUser dengan AWS SDK atau alat baris perintah](#)
 - [Gunakan CreateVirtualMfaDevice dengan AWS SDK atau alat baris perintah](#)

- [Gunakan DeactivateMfaDevice dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteAccessKey dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteAccountAlias dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteAccountPasswordPolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteGroup dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteGroupPolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteInstanceProfile dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteLoginProfile dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteOpenIdConnectProvider dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeletePolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeletePolicyVersion dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteRole dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteRolePermissionsBoundary dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteRolePolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteSAMLProvider dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteServerCertificate dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteServiceLinkedRole dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteSigningCertificate dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteUser dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteUserPermissionsBoundary dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteUserPolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteVirtualMfaDevice dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DetachGroupPolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DetachRolePolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DetachUserPolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan EnableMfaDevice dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GenerateCredentialReport dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GenerateServiceLastAccessedDetails dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetAccessKeyLastUsed dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetAccountAuthorizationDetails dengan AWS SDK atau alat baris perintah](#)

- [Gunakan GetAccountPasswordPolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetAccountSummary dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetContextKeysForCustomPolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetContextKeysForPrincipalPolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetCredentialReport dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetGroup dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetGroupPolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetInstanceProfile dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetLoginProfile dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetOpenIdConnectProvider dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetPolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetPolicyVersion dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetRole dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetRolePolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetSamlProvider dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetServerCertificate dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetServiceLastAccessedDetails dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetServiceLastAccessedDetailsWithEntities dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetServiceLinkedRoleDeletionStatus dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetUser dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetUserPolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListAccessKeys dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListAccountAliases dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListAttachedGroupPolicies dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListAttachedRolePolicies dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListAttachedUserPolicies dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListEntitiesForPolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListGroupPolicies dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListGroups dengan AWS SDK atau alat baris perintah](#)

- [Gunakan ListGroupsForUser dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListInstanceProfiles dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListInstanceProfilesForRole dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListMfaDevices dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListOpenIdConnectProviders dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListPolicies dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListPolicyVersions dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListRolePolicies dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListRoleTags dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListRoles dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListSAMLProviders dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListServerCertificates dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListSigningCertificates dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListUserPolicies dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListUserTags dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListUsers dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListVirtualMfaDevices dengan AWS SDK atau alat baris perintah](#)
- [Gunakan PutGroupPolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan PutRolePermissionsBoundary dengan AWS SDK atau alat baris perintah](#)
- [Gunakan PutRolePolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan PutUserPermissionsBoundary dengan AWS SDK atau alat baris perintah](#)
- [Gunakan PutUserPolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan RemoveClientIdFromOpenIdConnectProvider dengan AWS SDK atau alat baris perintah](#)
- [Gunakan RemoveRoleFromInstanceProfile dengan AWS SDK atau alat baris perintah](#)
- [Gunakan RemoveUserFromGroup dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ResyncMfaDevice dengan AWS SDK atau alat baris perintah](#)
- [Gunakan SetDefaultPolicyVersion dengan AWS SDK atau alat baris perintah](#)
- [Gunakan TagRole dengan AWS SDK atau alat baris perintah](#)
- [Gunakan TagUser dengan AWS SDK atau alat baris perintah](#)

- [Gunakan UntagRole dengan AWS SDK atau alat baris perintah](#)
- [Gunakan UntagUser dengan AWS SDK atau alat baris perintah](#)
- [Gunakan UpdateAccessKey dengan AWS SDK atau alat baris perintah](#)
- [Gunakan UpdateAccountPasswordPolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan UpdateAssumeRolePolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan UpdateGroup dengan AWS SDK atau alat baris perintah](#)
- [Gunakan UpdateLoginProfile dengan AWS SDK atau alat baris perintah](#)
- [Gunakan UpdateOpenIdConnectProviderThumbprint dengan AWS SDK atau alat baris perintah](#)
- [Gunakan UpdateRole dengan AWS SDK atau alat baris perintah](#)
- [Gunakan UpdateRoleDescription dengan AWS SDK atau alat baris perintah](#)
- [Gunakan UpdateSamlProvider dengan AWS SDK atau alat baris perintah](#)
- [Gunakan UpdateServerCertificate dengan AWS SDK atau alat baris perintah](#)
- [Gunakan UpdateSigningCertificate dengan AWS SDK atau alat baris perintah](#)
- [Gunakan UpdateUser dengan AWS SDK atau alat baris perintah](#)
- [Gunakan UploadServerCertificate dengan AWS SDK atau alat baris perintah](#)
- [Gunakan UploadSigningCertificate dengan AWS SDK atau alat baris perintah](#)
- [Skenario untuk IAM menggunakan AWS SDK](#)
 - [Membangun dan mengelola layanan tangguh menggunakan SDK AWS](#)
 - [Buat grup IAM dan tambahkan pengguna ke grup menggunakan SDK AWS](#)
 - [Buat pengguna IAM dan ambil peran dengan AWS STS menggunakan SDK AWS](#)
 - [Buat pengguna IAM read-only dan read-write menggunakan SDK AWS](#)
 - [Mengelola kunci akses IAM menggunakan SDK AWS](#)
 - [Mengelola kebijakan IAM menggunakan SDK AWS](#)
 - [Mengelola peran IAM menggunakan SDK AWS](#)
 - [Mengelola akun IAM Anda menggunakan SDK AWS](#)
 - [Kembalikan versi kebijakan IAM menggunakan SDK AWS](#)
 - [Bekerja dengan API Pembuat Kebijakan IAM menggunakan SDK AWS](#)
- [Contoh kode untuk AWS STS menggunakan AWS SDK](#)
 - [Tindakan untuk AWS STS menggunakan AWS SDK](#)

- [Gunakan AssumeRole dengan AWS SDK atau alat baris perintah](#)
- [Gunakan AssumeRoleWithWebIdentity dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DecodeAuthorizationMessage dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetFederationToken dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetSessionToken dengan AWS SDK atau alat baris perintah](#)
- [Skenario untuk AWS STS menggunakan AWS SDK](#)
 - [Asumsikan peran IAM yang membutuhkan token MFA AWS STS dengan menggunakan SDK AWS](#)
 - [Membangun URL dengan AWS STS untuk pengguna federasi menggunakan SDK AWS](#)
 - [Dapatkan token sesi yang memerlukan token MFA dengan AWS STS menggunakan SDK AWS](#)

Contoh kode untuk IAM menggunakan AWS SDK

Contoh kode berikut menunjukkan cara menggunakan IAM dengan kit pengembangan AWS perangkat lunak (SDK).

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Meskipun tindakan menunjukkan cara memanggil setiap fungsi layanan, Anda dapat melihat tindakan dalam konteks pada skenario yang terkait dan contoh lintas layanan.

Skenario adalah contoh kode yang menunjukkan cara untuk menyelesaikan tugas tertentu dengan memanggil beberapa fungsi dalam layanan yang sama.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga berisi informasi tentang cara memulai dan detail tentang versi SDK sebelumnya.

Memulai

Halo IAM

Contoh kode berikut menunjukkan bagaimana memulai menggunakan IAM.

.NET

AWS SDK for .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 [ListPolicies](#) di Referensi AWS SDK for .NET API.

C++

SDK for C++

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

Kode untuk file CMake MakeLists C.txt.

```
# 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)
  # 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 [ListPolicies](#) di Referensi AWS SDK for C++ API.

Go

SDK for 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() {
    sdkConfig, err := config.LoadDefaultConfig(context.TODO())
    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(context.TODO(), &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\t%v\n", *policy.PolicyName)
    }
}
}
```

- Untuk detail API, lihat [ListPolicies](#) di Referensi AWS SDK for Go API.

Java

SDK for 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 [ListPolicies](#) di 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) {
      page.Policies.forEach((p) => {
        console.log(`${p.PolicyName}`);
        policyCount++;
      });
    }
  }
  console.log(`Found ${policyCount} policies.`);
};
```

- Untuk detail API, lihat [ListPolicies](#) di Referensi AWS SDK for JavaScript 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<(), 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/ .rsiam-service-lib.

```

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 [ListPolicies](#) referensi AWS SDK for Rust API.

Contoh kode

- [Tindakan untuk IAM menggunakan AWS SDK](#)
 - [Gunakan AddClientIdToOpenIdConnectProvider dengan AWS SDK atau alat baris perintah](#)
 - [Gunakan AddRoleToInstanceProfile dengan AWS SDK atau alat baris perintah](#)
 - [Gunakan AddUserToGroup dengan AWS SDK atau alat baris perintah](#)
 - [Gunakan AttachGroupPolicy dengan AWS SDK atau alat baris perintah](#)
 - [Gunakan AttachRolePolicy dengan AWS SDK atau alat baris perintah](#)
 - [Gunakan AttachUserPolicy dengan AWS SDK atau alat baris perintah](#)
 - [Gunakan ChangePassword dengan AWS SDK atau alat baris perintah](#)
 - [Gunakan CreateAccessKey dengan AWS SDK atau alat baris perintah](#)
 - [Gunakan CreateAccountAlias dengan AWS SDK atau alat baris perintah](#)
 - [Gunakan CreateGroup dengan AWS SDK atau alat baris perintah](#)
 - [Gunakan CreateInstanceProfile dengan AWS SDK atau alat baris perintah](#)
 - [Gunakan CreateLoginProfile dengan AWS SDK atau alat baris perintah](#)
 - [Gunakan CreateOpenIdConnectProvider dengan AWS SDK atau alat baris perintah](#)
 - [Gunakan CreatePolicy dengan AWS SDK atau alat baris perintah](#)
 - [Gunakan CreatePolicyVersion dengan AWS SDK atau alat baris perintah](#)
 - [Gunakan CreateRole dengan AWS SDK atau alat baris perintah](#)

- [Gunakan CreateSAMLProvider dengan AWS SDK atau alat baris perintah](#)
- [Gunakan CreateServiceLinkedRole dengan AWS SDK atau alat baris perintah](#)
- [Gunakan CreateUser dengan AWS SDK atau alat baris perintah](#)
- [Gunakan CreateVirtualMfaDevice dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeactivateMfaDevice dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteAccessKey dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteAccountAlias dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteAccountPasswordPolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteGroup dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteGroupPolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteInstanceProfile dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteLoginProfile dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteOpenIdConnectProvider dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeletePolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeletePolicyVersion dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteRole dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteRolePermissionsBoundary dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteRolePolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteSAMLProvider dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteServerCertificate dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteServiceLinkedRole dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteSigningCertificate dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteUser dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteUserPermissionsBoundary dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteUserPolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteVirtualMfaDevice dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DetachGroupPolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DetachRolePolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DetachUserPolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan EnableMfaDevice dengan AWS SDK atau alat baris perintah](#)

- [Gunakan GenerateCredentialReport dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GenerateServiceLastAccessedDetails dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetAccessKeyLastUsed dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetAccountAuthorizationDetails dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetAccountPasswordPolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetAccountSummary dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetContextKeysForCustomPolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetContextKeysForPrincipalPolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetCredentialReport dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetGroup dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetGroupPolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetInstanceProfile dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetLoginProfile dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetOpenIdConnectProvider dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetPolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetPolicyVersion dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetRole dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetRolePolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetSamlProvider dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetServerCertificate dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetServiceLastAccessedDetails dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetServiceLastAccessedDetailsWithEntities dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetServiceLinkedRoleDeletionStatus dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetUser dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetUserPolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListAccessKeys dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListAccountAliases dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListAttachedGroupPolicies dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListAttachedRolePolicies dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListAttachedUserPolicies dengan AWS SDK atau alat baris perintah](#)

- [Gunakan ListEntitiesForPolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListGroupPolicies dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListGroups dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListGroupsForUser dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListInstanceProfiles dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListInstanceProfilesForRole dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListMfaDevices dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListOpenIdConnectProviders dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListPolicies dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListPolicyVersions dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListRolePolicies dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListRoleTags dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListRoles dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListSAMLProviders dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListServerCertificates dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListSigningCertificates dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListUserPolicies dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListUserTags dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListUsers dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListVirtualMfaDevices dengan AWS SDK atau alat baris perintah](#)
- [Gunakan PutGroupPolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan PutRolePermissionsBoundary dengan AWS SDK atau alat baris perintah](#)
- [Gunakan PutRolePolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan PutUserPermissionsBoundary dengan AWS SDK atau alat baris perintah](#)
- [Gunakan PutUserPolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan RemoveClientIdFromOpenIdConnectProvider dengan AWS SDK atau alat baris perintah](#)
- [Gunakan RemoveRoleFromInstanceProfile dengan AWS SDK atau alat baris perintah](#)
- [Gunakan RemoveUserFromGroup dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ResyncMfaDevice dengan AWS SDK atau alat baris perintah](#)

- [Gunakan SetDefaultPolicyVersion dengan AWS SDK atau alat baris perintah](#)
- [Gunakan TagRole dengan AWS SDK atau alat baris perintah](#)
- [Gunakan TagUser dengan AWS SDK atau alat baris perintah](#)
- [Gunakan UntagRole dengan AWS SDK atau alat baris perintah](#)
- [Gunakan UntagUser dengan AWS SDK atau alat baris perintah](#)
- [Gunakan UpdateAccessKey dengan AWS SDK atau alat baris perintah](#)
- [Gunakan UpdateAccountPasswordPolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan UpdateAssumeRolePolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan UpdateGroup dengan AWS SDK atau alat baris perintah](#)
- [Gunakan UpdateLoginProfile dengan AWS SDK atau alat baris perintah](#)
- [Gunakan UpdateOpenIdConnectProviderThumbprint dengan AWS SDK atau alat baris perintah](#)
- [Gunakan UpdateRole dengan AWS SDK atau alat baris perintah](#)
- [Gunakan UpdateRoleDescription dengan AWS SDK atau alat baris perintah](#)
- [Gunakan UpdateSamlProvider dengan AWS SDK atau alat baris perintah](#)
- [Gunakan UpdateServerCertificate dengan AWS SDK atau alat baris perintah](#)
- [Gunakan UpdateSigningCertificate dengan AWS SDK atau alat baris perintah](#)
- [Gunakan UpdateUser dengan AWS SDK atau alat baris perintah](#)
- [Gunakan UploadServerCertificate dengan AWS SDK atau alat baris perintah](#)
- [Gunakan UploadSigningCertificate dengan AWS SDK atau alat baris perintah](#)
- [Skenario untuk IAM menggunakan AWS SDK](#)
 - [Membangun dan mengelola layanan tangguh menggunakan SDK AWS](#)
 - [Buat grup IAM dan tambahkan pengguna ke grup menggunakan SDK AWS](#)
 - [Buat pengguna IAM dan ambil peran dengan AWS STS menggunakan SDK AWS](#)
 - [Buat pengguna IAM read-only dan read-write menggunakan SDK AWS](#)
 - [Mengelola kunci akses IAM menggunakan SDK AWS](#)
 - [Mengelola kebijakan IAM menggunakan SDK AWS](#)
 - [Mengelola peran IAM menggunakan SDK AWS](#)
 - [Mengelola akun IAM Anda menggunakan SDK AWS](#)
 - [Kembalikan versi kebijakan IAM menggunakan SDK AWS](#)
- [Bekerja dengan API Pembuat Kebijakan IAM menggunakan SDK AWS](#)

Tindakan untuk IAM menggunakan AWS SDK

Contoh kode berikut menunjukkan cara melakukan tindakan IAM individual dengan AWS SDK. Kutipan ini memanggil API IAM dan merupakan kutipan kode dari program yang lebih besar yang harus dijalankan dalam konteks. Setiap contoh menyertakan tautan ke GitHub, di mana Anda dapat menemukan instruksi untuk mengatur dan menjalankan kode.

Contoh berikut hanya mencakup tindakan yang paling umum digunakan. Untuk daftar lengkapnya, lihat [Referensi API AWS Identity and Access Management \(IAM\)](#).

Contoh-contoh

- [Gunakan AddClientIdToOpenIdConnectProvider dengan AWS SDK atau alat baris perintah](#)
- [Gunakan AddRoleToInstanceProfile dengan AWS SDK atau alat baris perintah](#)
- [Gunakan AddUserToGroup dengan AWS SDK atau alat baris perintah](#)
- [Gunakan AttachGroupPolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan AttachRolePolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan AttachUserPolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ChangePassword dengan AWS SDK atau alat baris perintah](#)
- [Gunakan CreateAccessKey dengan AWS SDK atau alat baris perintah](#)
- [Gunakan CreateAccountAlias dengan AWS SDK atau alat baris perintah](#)
- [Gunakan CreateGroup dengan AWS SDK atau alat baris perintah](#)
- [Gunakan CreateInstanceProfile dengan AWS SDK atau alat baris perintah](#)
- [Gunakan CreateLoginProfile dengan AWS SDK atau alat baris perintah](#)
- [Gunakan CreateOpenIdConnectProvider dengan AWS SDK atau alat baris perintah](#)
- [Gunakan CreatePolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan CreatePolicyVersion dengan AWS SDK atau alat baris perintah](#)
- [Gunakan CreateRole dengan AWS SDK atau alat baris perintah](#)
- [Gunakan CreateSAMLProvider dengan AWS SDK atau alat baris perintah](#)
- [Gunakan CreateServiceLinkedRole dengan AWS SDK atau alat baris perintah](#)
- [Gunakan CreateUser dengan AWS SDK atau alat baris perintah](#)
- [Gunakan CreateVirtualMfaDevice dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeactivateMfaDevice dengan AWS SDK atau alat baris perintah](#)

- [Gunakan DeleteAccessKey dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteAccountAlias dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteAccountPasswordPolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteGroup dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteGroupPolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteInstanceProfile dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteLoginProfile dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteOpenIdConnectProvider dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeletePolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeletePolicyVersion dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteRole dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteRolePermissionsBoundary dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteRolePolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteSAMLProvider dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteServerCertificate dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteServiceLinkedRole dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteSigningCertificate dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteUser dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteUserPermissionsBoundary dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteUserPolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DeleteVirtualMfaDevice dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DetachGroupPolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DetachRolePolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DetachUserPolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan EnableMfaDevice dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GenerateCredentialReport dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GenerateServiceLastAccessedDetails dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetAccessKeyLastUsed dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetAccountAuthorizationDetails dengan AWS SDK atau alat baris perintah](#)

- [Gunakan `GetAccountPasswordPolicy` dengan AWS SDK atau alat baris perintah](#)
- [Gunakan `GetAccountSummary` dengan AWS SDK atau alat baris perintah](#)
- [Gunakan `GetContextKeysForCustomPolicy` dengan AWS SDK atau alat baris perintah](#)
- [Gunakan `GetContextKeysForPrincipalPolicy` dengan AWS SDK atau alat baris perintah](#)
- [Gunakan `GetCredentialReport` dengan AWS SDK atau alat baris perintah](#)
- [Gunakan `GetGroup` dengan AWS SDK atau alat baris perintah](#)
- [Gunakan `GetGroupPolicy` dengan AWS SDK atau alat baris perintah](#)
- [Gunakan `GetInstanceProfile` dengan AWS SDK atau alat baris perintah](#)
- [Gunakan `GetLoginProfile` dengan AWS SDK atau alat baris perintah](#)
- [Gunakan `GetOpenIdConnectProvider` dengan AWS SDK atau alat baris perintah](#)
- [Gunakan `GetPolicy` dengan AWS SDK atau alat baris perintah](#)
- [Gunakan `GetPolicyVersion` dengan AWS SDK atau alat baris perintah](#)
- [Gunakan `GetRole` dengan AWS SDK atau alat baris perintah](#)
- [Gunakan `GetRolePolicy` dengan AWS SDK atau alat baris perintah](#)
- [Gunakan `GetSamlProvider` dengan AWS SDK atau alat baris perintah](#)
- [Gunakan `GetServerCertificate` dengan AWS SDK atau alat baris perintah](#)
- [Gunakan `GetServiceLastAccessedDetails` dengan AWS SDK atau alat baris perintah](#)
- [Gunakan `GetServiceLastAccessedDetailsWithEntities` dengan AWS SDK atau alat baris perintah](#)
- [Gunakan `GetServiceLinkedRoleDeletionStatus` dengan AWS SDK atau alat baris perintah](#)
- [Gunakan `GetUser` dengan AWS SDK atau alat baris perintah](#)
- [Gunakan `GetUserPolicy` dengan AWS SDK atau alat baris perintah](#)
- [Gunakan `ListAccessKeys` dengan AWS SDK atau alat baris perintah](#)
- [Gunakan `ListAccountAliases` dengan AWS SDK atau alat baris perintah](#)
- [Gunakan `ListAttachedGroupPolicies` dengan AWS SDK atau alat baris perintah](#)
- [Gunakan `ListAttachedRolePolicies` dengan AWS SDK atau alat baris perintah](#)
- [Gunakan `ListAttachedUserPolicies` dengan AWS SDK atau alat baris perintah](#)
- [Gunakan `ListEntitiesForPolicy` dengan AWS SDK atau alat baris perintah](#)
- [Gunakan `ListGroupPolicies` dengan AWS SDK atau alat baris perintah](#)
- [Gunakan `ListGroups` dengan AWS SDK atau alat baris perintah](#)

- [Gunakan ListGroupsForUser dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListInstanceProfiles dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListInstanceProfilesForRole dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListMfaDevices dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListOpenIdConnectProviders dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListPolicies dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListPolicyVersions dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListRolePolicies dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListRoleTags dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListRoles dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListSAMLProviders dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListServerCertificates dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListSigningCertificates dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListUserPolicies dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListUserTags dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListUsers dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ListVirtualMfaDevices dengan AWS SDK atau alat baris perintah](#)
- [Gunakan PutGroupPolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan PutRolePermissionsBoundary dengan AWS SDK atau alat baris perintah](#)
- [Gunakan PutRolePolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan PutUserPermissionsBoundary dengan AWS SDK atau alat baris perintah](#)
- [Gunakan PutUserPolicy dengan AWS SDK atau alat baris perintah](#)
- [Gunakan RemoveClientIdFromOpenIdConnectProvider dengan AWS SDK atau alat baris perintah](#)
- [Gunakan RemoveRoleFromInstanceProfile dengan AWS SDK atau alat baris perintah](#)
- [Gunakan RemoveUserFromGroup dengan AWS SDK atau alat baris perintah](#)
- [Gunakan ResyncMfaDevice dengan AWS SDK atau alat baris perintah](#)
- [Gunakan SetDefaultPolicyVersion dengan AWS SDK atau alat baris perintah](#)
- [Gunakan TagRole dengan AWS SDK atau alat baris perintah](#)
- [Gunakan TagUser dengan AWS SDK atau alat baris perintah](#)
- [Gunakan UntagRole dengan AWS SDK atau alat baris perintah](#)

- [Gunakan `UntagUser` dengan AWS SDK atau alat baris perintah](#)
- [Gunakan `UpdateAccessKey` dengan AWS SDK atau alat baris perintah](#)
- [Gunakan `UpdateAccountPasswordPolicy` dengan AWS SDK atau alat baris perintah](#)
- [Gunakan `UpdateAssumeRolePolicy` dengan AWS SDK atau alat baris perintah](#)
- [Gunakan `UpdateGroup` dengan AWS SDK atau alat baris perintah](#)
- [Gunakan `UpdateLoginProfile` dengan AWS SDK atau alat baris perintah](#)
- [Gunakan `UpdateOpenIdConnectProviderThumbprint` dengan AWS SDK atau alat baris perintah](#)
- [Gunakan `UpdateRole` dengan AWS SDK atau alat baris perintah](#)
- [Gunakan `UpdateRoleDescription` dengan AWS SDK atau alat baris perintah](#)
- [Gunakan `UpdateSamlProvider` dengan AWS SDK atau alat baris perintah](#)
- [Gunakan `UpdateServerCertificate` dengan AWS SDK atau alat baris perintah](#)
- [Gunakan `UpdateSigningCertificate` dengan AWS SDK atau alat baris perintah](#)
- [Gunakan `UpdateUser` dengan AWS SDK atau alat baris perintah](#)
- [Gunakan `UploadServerCertificate` dengan AWS SDK atau alat baris perintah](#)
- [Gunakan `UploadSigningCertificate` dengan AWS SDK atau alat baris perintah](#)

Gunakan **`AddClientIdToOpenIdConnectProvider`** dengan AWS SDK atau alat baris perintah

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 [AddClientIdToOpenIdConnectProvider](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [AddClientIdToOpenIdConnectProvider](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **AddRoleToInstanceProfile** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan `AddRoleToInstanceProfile`.

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 di instans Amazon EC2](#) dalam AWS Panduan Pengguna IAM.

- Untuk detail API, lihat [AddRoleToInstanceProfile](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [AddRoleToInstanceProfile](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **AddUserToGroup** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan `AddUserToGroup`.

Contoh-contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan di dalam konteks. Anda dapat melihat tindakan ini dalam konteks pada contoh kode berikut:

- [Buat grup dan tambahkan pengguna](#)

.NET

AWS SDK for .NET

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
/// <summary>
/// Add an existing IAM user to an existing IAM group.
/// </summary>
/// <param name="userName">The username of the user to add.</param>
/// <param name="groupName">The name of the group to add the user to.</param>
/// <returns>A Boolean value indicating the success of the action.</returns>
public async Task<bool> AddUserToGroupAsync(string userName, string
groupName)
{
    var response = await _IAMService.AddUserToGroupAsync(new
AddUserToGroupRequest
    {
        GroupName = groupName,
        UserName = userName,
    });

    return response.HttpStatusCode == HttpStatusCode.OK;
}
```

- Untuk detail API, lihat [AddUserToGroup](#) di Referensi AWS SDK for .NET API.

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 [AddUserToGroup](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini menambahkan nama pengguna **Bob** ke grup bernama **Admins**.

```
Add-IAMUserToGroup -UserName "Bob" -GroupName "Admins"
```

- Untuk detail API, lihat [AddUserToGroup](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **AttachGroupPolicy** dengan AWS SDK atau alat baris perintah

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 [AttachGroupPolicy](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [AttachGroupPolicy](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **AttachRolePolicy** dengan AWS SDK atau alat baris perintah

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:

- [Buat grup dan tambahkan pengguna](#)
- [Buat pengguna dan ambil peran](#)
- [Kelola peran](#)

.NET

AWS SDK for .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 [AttachRolePolicy](#) di Referensi AWS SDK for .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 role.
#
# 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 getopt 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
```

```

fi

echo "$response"

return 0
}

```

- Untuk detail API, lihat [AttachRolePolicy](#) di Referensi AWS CLI Perintah.

C++

SDK for 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 [AttachRolePolicy](#) di Referensi AWS SDK for 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 [AttachRolePolicy](#) di Referensi AWS CLI Perintah.

Go

SDK for Go V2

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
// 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(policyArn string, roleName string)  
  error {  
  _, err := wrapper.IamClient.AttachRolePolicy(context.TODO(),  
    &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 [AttachRolePolicy](#) di Referensi AWS SDK for Go API.

Java

SDK for 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 [AttachRolePolicy](#) di 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 for JavaScript](#).
- Untuk detail API, lihat [AttachRolePolicy](#) di Referensi AWS SDK for 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 for JavaScript](#).
- Untuk detail API, lihat [AttachRolePolicy](#) di Referensi AWS SDK for JavaScript API.

Kotlin

SDK for 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 { 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 [AttachRolePolicy](#) di AWS SDK untuk referensi API Kotlin.

PHP

SDK for 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 [AttachRolePolicy](#) di Referensi AWS SDK for PHP API.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini melampirkan kebijakan AWS terkelola bernama **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 [AttachRolePolicy](#) di Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK for 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 [AttachRolePolicy](#) di AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK for 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 [AttachRolePolicy](#) di Referensi AWS SDK for 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 [AttachRolePolicy](#) referensi AWS SDK for Rust API.

Swift

SDK for Swift

Note

Ini adalah dokumentasi prarilis untuk SDK dalam rilis pratinjau. Dokumentasi ini dapat berubah.

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
public func attachRolePolicy(role: String, policyArn: String) async throws {
    let input = AttachRolePolicyInput(
        policyArn: policyArn,
        roleName: role
    )
    do {
        _ = try await client.attachRolePolicy(input: input)
    } catch {
        throw error
    }
}
```

- Untuk detail API, lihat referensi [AttachRolePolicy AWSSDK](#) untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **AttachUserPolicy** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan `AttachUserPolicy`.

Contoh-contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan di dalam konteks. Anda dapat melihat tindakan ini dalam konteks pada 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 [AttachUserPolicy](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [AttachUserPolicy](#) di Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK for 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 [AttachUserPolicy](#) di AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK for 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 [AttachUserPolicy](#) di Referensi AWS SDK for 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 [AttachUserPolicy](#) referensi AWS SDK for Rust API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **ChangePassword** dengan AWS SDK atau alat baris perintah

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 [ChangePassword](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [ChangePassword](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **CreateAccessKey** dengan AWS SDK atau alat baris perintah

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:

- [Buat grup dan tambahkan pengguna](#)
- [Buat pengguna dan ambil peran](#)
- [Buat pengguna read-only dan read-write](#)
- [Kelola kunci akses](#)

.NET

AWS SDK for .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 [CreateAccessKey](#) di Referensi AWS SDK for .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 getopt 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 [CreateAccessKey](#) di Referensi AWS CLI Perintah.

C++

SDK for 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 [CreateAccessKey](#) di Referensi AWS SDK for 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 bernama Bob.

```
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 [CreateAccessKey](#) di Referensi AWS CLI Perintah.

Go

SDK for Go V2

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
// 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(userName string)
(*types.AccessKey, error) {
    var key *types.AccessKey
    result, err := wrapper.IamClient.CreateAccessKey(context.TODO(),
&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 [CreateAccessKey](#) di Referensi AWS SDK for Go API.

Java

SDK for 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 [CreateAccessKey](#) di 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 for JavaScript](#).
- Untuk detail API, lihat [CreateAccessKey](#) di Referensi AWS SDK for 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 for JavaScript](#).
- Untuk detail API, lihat [CreateAccessKey](#) di Referensi AWS SDK for JavaScript API.

Kotlin

SDK for 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 { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createAccessKey(request)
        return response.accessKey?.accessKeyId.toString()
    }
}
```

- Untuk detail API, lihat [CreateAccessKey](#) di AWS SDK untuk referensi API Kotlin.

PowerShell

Alat untuk PowerShell

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 [CreateAccessKey](#) di Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK for 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 [CreateAccessKey](#) di AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK for 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 => e
  @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 => e
  @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"
  )
end
```

```

    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
end

```

- Untuk detail API, lihat [CreateAccessKey](#) di Referensi AWS SDK for 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 [CreateAccessKey](#) referensi AWS SDK for Rust API.

Swift

SDK for Swift

Note

Ini adalah dokumentasi prarilis untuk SDK dalam rilis pratinjau. Dokumentasi ini dapat berubah.

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```

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 {
        throw error
    }
}
```

- Untuk detail API, lihat referensi [CreateAccessKey AWSSDK](#) untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **CreateAccountAlias** dengan AWS SDK atau alat baris perintah


Contoh kode berikut menunjukkan cara menggunakan `CreateAccountAlias`.

Contoh-contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan di dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- [Kelola akun Anda](#)

C++

SDK for 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 [CreateAccountAlias](#) di Referensi AWS SDK for 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 [CreateAccountAlias](#) di Referensi AWS CLI Perintah.

Java

SDK for 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 [CreateAccountAlias](#) di 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 for JavaScript](#).
- Untuk detail API, lihat [CreateAccountAlias](#) di Referensi AWS SDK for 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 for JavaScript](#).
- Untuk detail API, lihat [CreateAccountAlias](#) di Referensi AWS SDK for JavaScript API.

Kotlin

SDK for 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 { region = "AWS_GLOBAL" }.use { iamClient ->  
        iamClient.createAccountAlias(request)  
        println("Successfully created account alias named $alias")  
    }  
}
```

- Untuk detail API, lihat [CreateAccountAlias](#) di AWS SDK untuk referensi API Kotlin.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengubah alias akun untuk AWS **mycompanyaws** akun Anda. Alamat halaman logon pengguna mengarah ke <https://mycompanyaws.signin.aws.amazon.com/console>. URL asli menggunakan nomor ID akun Anda, bukan alias (<https://>

<accountidnumber>.signin.aws.amazon.com/console) terus berfungsi. Namun, URL berbasis alias yang ditentukan sebelumnya berhenti berfungsi.

```
New-IAMAccountAlias -AccountAlias mycompanyaws
```

- Untuk detail API, lihat [CreateAccountAlias](#) di Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK for 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 [CreateAccountAlias](#) di AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK for 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
  end
end
```



```
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 [CreateAccountAlias](#) di Referensi AWS SDK for Ruby API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **CreateGroup** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan `CreateGroup`.

Contoh-contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan di dalam konteks. Anda dapat melihat tindakan ini dalam konteks pada contoh kode berikut:

- [Buat grup dan tambahkan pengguna](#)

.NET

AWS SDK for .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 group.
/// </summary>
/// <param name="groupName">The name to give the IAM group.</param>
/// <returns>The IAM group that was created.</returns>
public async Task<Group> CreateGroupAsync(string groupName)
{
    var response = await _IAMService.CreateGroupAsync(new CreateGroupRequest
{ GroupName = groupName });
    return response.Group;
}
```

- Untuk detail API, lihat [CreateGroup](#) di Referensi AWS SDK for .NET API.

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": "AIDGPMS9R04H3FEXAMPLE",
    "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 [CreateGroup](#) di 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 [CreateGroup](#) di Referensi AWS SDK for JavaScript API.

PowerShell

Alat untuk PowerShell

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 [CreateGroup](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **CreateInstanceProfile** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan `CreateInstanceProfile`.

Contoh-contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan di dalam konteks. Anda dapat melihat tindakan ini dalam konteks pada contoh kode berikut:

- [Membangun dan mengelola layanan yang tangguh](#)

.NET

AWS SDK for .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 [CreateInstanceProfile](#) di Referensi AWS SDK for .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": {  
    "InstanceId": "AIPAJMBC7DLSPEXAMPLE",  
    "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 di instans Amazon EC2](#) dalam AWS Panduan Pengguna IAM.

- Untuk detail API, lihat [CreateInstanceProfile](#) di 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 [CreateInstanceProfile](#) di Referensi AWS SDK for JavaScript API.

PowerShell

Alat untuk PowerShell

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 [CreateInstanceProfile](#) di Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK for 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 AutoScaler:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix,
        inst_type,
        ami_param,
        autoscaling_client,
        ec2_client,
        ssm_client,
        iam_client,
    ):
        """
        :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
        self.launch_template_name = f"{resource_prefix}-template"
        self.group_name = f"{resource_prefix}-group"
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"
        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"
        self.key_pair_name = f"{resource_prefix}-key-pair"

    def create_instance_profile(
        self, policy_file, policy_name, role_name, profile_name,
aws_managed_policies=()
    ):
        """
        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",
            }
        ],
    }
    with open(policy_file) as file:
        instance_policy_doc = file.read()

    policy_arn = None
    try:
        pol_response = self.iam_client.create_policy(
            PolicyName=policy_name, PolicyDocument=instance_policy_doc
        )
        policy_arn = pol_response["Policy"]["Arn"]
        log.info("Created policy with ARN %s.", policy_arn)
    except ClientError as err:
        if err.response["Error"]["Code"] == "EntityAlreadyExists":
            log.info("Policy %s already exists, nothing to do.", policy_name)
            list_pol_response = self.iam_client.list_policies(Scope="Local")
            for pol in list_pol_response["Policies"]:
                if pol["PolicyName"] == policy_name:
                    policy_arn = pol["Arn"]
                    break
        if policy_arn is None:
            raise AutoScalerError(f"Couldn't create policy {policy_name}:
{err}")

    try:
        self.iam_client.create_role(
            RoleName=role_name,
            AssumeRolePolicyDocument=json.dumps(assume_role_doc)

```

```
    )
    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("Created role %s and attached policy %s.", role_name,
policy_arn)
    except ClientError as err:
        if err.response["Error"]["Code"] == "EntityAlreadyExists":
            log.info("Role %s already exists, nothing to do.", role_name)
        else:
            raise AutoScalerError(f"Couldn't create role {role_name}: {err}")

    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
            )
        else:
            raise AutoScalerError(
                f"Couldn't create profile {profile_name} and attach it to
role\n"
                f"{role_name}: {err}")
```

```
)  
return profile_arn
```

- Untuk detail API, lihat [CreateInstanceProfile](#) di AWS SDK for Python (Boto3) Referensi API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **CreateLoginProfile** dengan AWS SDK atau alat baris perintah

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 yang disebut `create-login-profile.json` yang dapat Anda gunakan untuk mengisi informasi untuk perintah berikutnya. `create-login-profile` Sebagai 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 yang disebut `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 untuk, 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 [CreateLoginProfile](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [CreateLoginProfile](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **CreateOpenIdConnectProvider** dengan AWS SDK atau alat baris perintah

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` Sebagai contoh:

```
{  
  "Url": "https://server.example.com",  
  "ClientIDList": [  
    "example-application-ID"  
  ],  
  "ThumbprintList": [  
    "c3768084dfb3d2b68b7897bf5f565da8eEXAMPLE"  
  ]  
}
```

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 [CreateOpenIdConnectProvider](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 cap jempol, ikuti langkah-langkah di <http://docs.aws.amazon.com/IAM/latest/UserGuide/-thumbprint.html>.
identity-providers-oidc-obtain

```
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 [CreateOpenIdConnectProvider](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **CreatePolicy** dengan AWS SDK atau alat baris perintah

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:

- [Buat grup dan tambahkan pengguna](#)
- [Buat pengguna dan ambil peran](#)

- [Buat pengguna read-only dan read-write](#)
- [Kelola kebijakan](#)
- [Bekerja dengan API Pembuat Kebijakan IAM](#)

.NET

AWS SDK for .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 [CreatePolicy](#) di Referensi AWS SDK for .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 getopt 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 [CreatePolicy](#) di Referensi AWS CLI Perintah.

C++

SDK for 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 [CreatePolicy](#) di Referensi AWS SDK for C++ API.

CLI

AWS CLI

Contoh 1: Untuk membuat kebijakan terkelola pelanggan

Perintah berikut membuat kebijakan terkelola pelanggan bernama `my-policy`.

```
aws iam create-policy \  
  --policy-name my-policy \  
  --policy-document file://policy
```

File tersebut `policy` adalah dokumen JSON di folder saat ini yang memberikan akses baca saja ke `shared` folder dalam bucket Amazon S3 bernama `my-bucket`

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Effect": "Allow",  
      "Action": [  
        "s3:Get*",  
        "s3:List*"  
      ],  
      "Resource": [  
        "arn:aws:s3:::my-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:

```
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 my-bucket"
```

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 `my-bucket`

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Effect": "Allow",  
      "Action": [  
        "s3:ListBucket*",  
        "s3:PutBucket*",  
        "s3:GetBucket*"  
      ],  
      "Resource": [  
        "arn:aws:s3:::my-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 Kebijakan Berbasis Identifikasi, lihat Kebijakan berbasis identitas dan kebijakan berbasis sumber daya di [Panduan Pengguna IAM.AWS](#)

Contoh 3: Membuat kebijakan terkelola pelanggan dengan tag

Perintah berikut membuat kebijakan terkelola pelanggan bernama my-policy dengan tag. Contoh ini menggunakan flag --tags parameter dengan tag berformat JSON 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-policy \
  --policy-name my-policy \
  --policy-document file://policy.json \
  --tags '{"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 my-bucket

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:ListBucket*",
        "s3:PutBucket*",
        "s3:GetBucket*"
      ],
      "Resource": [
        "arn:aws:s3:::my-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 [CreatePolicy](#) di Referensi AWS CLI Perintah.

Go

SDK for Go V2

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
// 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
}

// 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(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(context.TODO(),
&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 [CreatePolicy](#) di Referensi AWS SDK for Go API.

Java

SDK for 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 [CreatePolicy](#) di 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 for JavaScript](#).
- Untuk detail API, lihat [CreatePolicy](#) di Referensi AWS SDK for 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 for JavaScript](#).
- Untuk detail API, lihat [CreatePolicy](#) di Referensi AWS SDK for JavaScript API.

Kotlin

SDK for 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\": \"*\":" +
    "    }" +
    "  ]" +
    "}"
```

```

val request = CreatePolicyRequest {
    policyName = policyNameVal
    policyDocument = policyDocumentVal
}

IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
    val response = iamClient.createPolicy(request)
    return response.policy?.arn.toString()
}
}

```

- Untuk detail API, lihat [CreatePolicy](#) di AWS SDK untuk referensi API Kotlin.

PHP

SDK for 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";

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 [CreatePolicy](#) di Referensi AWS SDK for PHP API.

PowerShell

Alat untuk PowerShell

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 [CreatePolicy](#) di Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK for 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::my-bucket/*' to allow actions on all
    objects
                        in the bucket named 'my-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 [CreatePolicy](#) di AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK for 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
end

```

- Untuk detail API, lihat [CreatePolicy](#) di Referensi AWS SDK for 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 [CreatePolicy](#) referensi AWS SDK for Rust API.

Swift

SDK for Swift

Note

Ini adalah dokumentasi prarilis untuk SDK dalam rilis pratinjau. Dokumentasi ini dapat berubah.

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
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 {
```

```
        throw error
    }
}
```

- Untuk detail API, lihat referensi [CreatePolicy AWSSDK](#) untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **CreatePolicyVersion** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan `CreatePolicyVersion`.

Contoh-contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan di dalam konteks. Anda dapat melihat tindakan ini dalam konteks pada 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 [CreatePolicyVersion](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 tersebut 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	VersionId	Document	IsDefaultVersion
-----	-----	-----	-----
4/15/2015 10:54:54 AM	v2		True

- Untuk detail API, lihat [CreatePolicyVersion](#) di Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK for Python (Boto3)

Note

Ada lebih banyak tentang GitHub. Temukan contoh selengkapnya 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 [CreatePolicyVersion](#) di AWS SDK for Python (Boto3) Referensi API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **CreateRole** dengan AWS SDK atau alat baris perintah

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:

- [Buat grup dan tambahkan pengguna](#)
- [Buat pengguna dan ambil peran](#)
- [Kelola peran](#)

.NET

AWS SDK for .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 [CreateRole](#) di Referensi AWS SDK for .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 getopt 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 [CreateRole](#) di Referensi AWS CLI Perintah.

C++

SDK for 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 [CreateRole](#) di Referensi AWS SDK for 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 file `test-role-trust-policy.json`. (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 bendera `--tags` parameter dengan tag berformat JSON 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 [CreateRole](#) di Referensi AWS CLI Perintah.

Go

SDK for Go V2

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```

// 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(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(context.TODO(),
&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 [CreateRole](#) di Referensi AWS SDK for Go API.

Java

SDK for 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 [CreateRole](#) di 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 [CreateRole](#) di Referensi AWS SDK for JavaScript API.

PHP

SDK for 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 [CreateRole](#) di Referensi AWS SDK for PHP API.

PowerShell

Alat untuk PowerShell

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%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 [CreateRole](#) di Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK for 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 [CreateRole](#) di AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK for 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 [CreateRole](#) di Referensi AWS SDK for 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 [CreateRole](#) referensi AWS SDK for Rust API.

Swift

SDK for Swift

Note

Ini adalah dokumentasi prarilis untuk SDK dalam rilis pratinjau. Dokumentasi ini dapat berubah.

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
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 {
    throw error
  }
}
```

- Untuk detail API, lihat referensi [CreateRole AWSSDK](#) untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **CreateSAMLProvider** dengan AWS SDK atau alat baris perintah

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 "fs";
import * as path from "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 for JavaScript

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini membuat entitas penyedia SAML baru di IAM. Ini dinamai **MySAMLProvider** dan dijelaskan oleh dokumen metadata SAML yang ditemukan dalam file **SAMLMetaData.xml**, yang diunduh secara terpisah dari situs web penyedia layanan SAML.

```
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 Cmdlet](#).AWS Tools for PowerShell

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **CreateServiceLinkedRole** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan `CreateServiceLinkedRole`.

.NET

AWS SDK for .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 [CreateServiceLinkedRole](#) di Referensi AWS SDK for .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 [CreateServiceLinkedRole](#) di Referensi AWS CLI Perintah.

Go

SDK for Go V2

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
// 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(serviceName string,
description string) (*types.Role, error) {
    var role *types.Role
    result, err := wrapper.IamClient.CreateServiceLinkedRole(context.TODO(),
&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 [CreateServiceLinkedRole](#) di Referensi AWS SDK for 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, 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,
  });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Untuk detail API, lihat [CreateServiceLinkedRole](#) di Referensi AWS SDK for JavaScript API.

PHP

SDK for 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 [CreateServiceLinkedRole](#) di Referensi AWS SDK for PHP API.

PowerShell

Alat untuk PowerShell

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 [CreateServiceLinkedRole](#) di Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK for 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 [CreateServiceLinkedRole](#) di AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK for 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 [CreateServiceLinkedRole](#) di Referensi AWS SDK for 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 [CreateServiceLinkedRole](#) referensi AWS SDK for Rust API.

Swift

SDK for Swift

Note

Ini adalah dokumentasi prarilis untuk SDK dalam rilis pratinjau. Dokumentasi ini dapat berubah.

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
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 {
        throw error
    }
}
```

- Untuk detail API, lihat referensi [CreateServiceLinkedRole AWSSDK](#) untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **CreateUser** dengan AWS SDK atau alat baris perintah

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:

- [Buat grup dan tambahkan pengguna](#)

- [Buat pengguna dan ambil peran](#)
- [Buat pengguna read-only dan read-write](#)

.NET

AWS SDK for .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 [CreateUser](#) di Referensi AWS SDK for .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 getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function iam_create_user"
```

```
    echo "Creates an WS 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 [CreateUser](#) di Referensi AWS CLI Perintah.

C++

SDK for 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 [CreateUser](#) di Referensi AWS SDK for 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 [pengidentifikasi 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 tag berformat JSON 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 AmazonS3. FullAccess

```
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 [Batas izin untuk entitas IAM](#) di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat [CreateUser](#) di Referensi AWS CLI Perintah.

Go

SDK for Go V2

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
// 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(userName string) (*types.User, error) {
    var user *types.User
    result, err := wrapper.IamClient.CreateUser(context.TODO(),
        &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 [CreateUser](#) di Referensi AWS SDK for Go API.

Java

SDK for 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 [CreateUser](#) di 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 for JavaScript](#).
- Untuk detail API, lihat [CreateUser](#) di Referensi AWS SDK for 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 for JavaScript](#).
- Untuk detail API, lihat [CreateUser](#) di Referensi AWS SDK for JavaScript API.

Kotlin

SDK for 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 { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createUser(request)
        return response.user?.userName
    }
}
```

- Untuk detail API, lihat [CreateUser](#) di AWS SDK untuk referensi API Kotlin.

PHP

SDK for 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 [CreateUser](#) di Referensi AWS SDK for PHP API.

PowerShell

Alat untuk PowerShell

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 [CreateUser](#) di Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK for 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 [CreateUser](#) di AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK for 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 [CreateUser](#) di Referensi AWS SDK for 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 [CreateUser](#) referensi AWS SDK for Rust API.

Swift

SDK for Swift

Note

Ini adalah dokumentasi prarilis untuk SDK dalam rilis pratinjau. Dokumentasi ini dapat berubah.

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
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 {
        throw error
    }
}
```

- Untuk detail API, lihat referensi [CreateUser AWSSDK](#) untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **CreateVirtualMfaDevice** dengan AWS SDK atau alat baris perintah

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 menemukannya 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 [CreateVirtualMfaDevice](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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
CZWZMCQNW4DEXAMPLE3V0UGXJFZYSUW7EXAMPLECR4NJFD65GX2SLUDW2EXAMPLE
```

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 nilai StringSeed Base32 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 [CreateVirtualMfaDevice](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan `DeactivateMfaDevice` dengan AWS SDK atau alat baris perintah

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 [DeactivateMfaDevice](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [DeactivateMfaDevice](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **DeleteAccessKey** dengan AWS SDK atau alat baris perintah

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:

- [Buat grup dan tambahkan pengguna](#)
- [Buat pengguna dan ambil peran](#)
- [Buat pengguna read-only dan read-write](#)
- [Kelola kunci akses](#)

.NET

AWS SDK for .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 [DeleteAccessKey](#) di Referensi AWS SDK for .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 getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function iam_delete_access_key"
        echo "Deletes an WS 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 getopt "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 [DeleteAccessKey](#) di Referensi AWS CLI Perintah.

C++

SDK for 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 [DeleteAccessKey](#) di Referensi AWS SDK for 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 AKIDPMS9R04H3FEXAMPLE \
  --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 [DeleteAccessKey](#) di Referensi AWS CLI Perintah.

Go

SDK for Go V2

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
// 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(userName string, keyId string) error {
    _, err := wrapper.IamClient.DeleteAccessKey(context.TODO(),
        &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 [DeleteAccessKey](#) di Referensi AWS SDK for Go API.

Java

SDK for 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 [DeleteAccessKey](#) di 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 for JavaScript](#).
- Untuk detail API, lihat [DeleteAccessKey](#) di Referensi AWS SDK for 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 for JavaScript](#).
- Untuk detail API, lihat [DeleteAccessKey](#) di Referensi AWS SDK for JavaScript API.

Kotlin

SDK for 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 { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.deleteAccessKey(request)
        println("Successfully deleted access key $accessKey from $userNameVal")
    }
}
```

- Untuk detail API, lihat [DeleteAccessKey](#) di AWS SDK untuk referensi API Kotlin.

PowerShell

Alat untuk PowerShell

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 [DeleteAccessKey](#) di Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK for 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 [DeleteAccessKey](#) di AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK for 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 => e
    @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 => e
    @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}")
  end
end
```

```
    false
  end
end
```

- Untuk detail API, lihat [DeleteAccessKey](#) di Referensi AWS SDK for 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 [DeleteAccessKey](#) referensi AWS SDK for Rust API.

Swift

SDK for Swift

Note

Ini adalah dokumentasi prarilis untuk SDK dalam rilis pratinjau. Dokumentasi ini dapat berubah.

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
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 {
        throw error
    }
}
```

- Untuk detail API, lihat referensi [DeleteAccessKey AWSSDK](#) untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **DeleteAccountAlias** dengan AWS SDK atau alat baris perintah


Contoh kode berikut menunjukkan cara menggunakan `DeleteAccountAlias`.

Contoh-contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan di dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- [Kelola akun Anda](#)

C++

SDK for 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 [DeleteAccountAlias](#) di Referensi AWS SDK for 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 [DeleteAccountAlias](#) di Referensi AWS CLI Perintah.

Java

SDK for 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 [DeleteAccountAlias](#) di 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 for JavaScript](#).
- Untuk detail API, lihat [DeleteAccountAlias](#) di Referensi AWS SDK for 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 for JavaScript](#).
- Untuk detail API, lihat [DeleteAccountAlias](#) di Referensi AWS SDK for JavaScript API.

Kotlin

SDK for 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 { region = "AWS_GLOBAL" }.use { iamClient ->  
        iamClient.deleteAccountAlias(request)  
        println("Successfully deleted account alias $alias")  
    }  
}
```

- Untuk detail API, lihat [DeleteAccountAlias](#) di AWS SDK untuk referensi API Kotlin.

PowerShell

Alat untuk PowerShell

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. Sebagai gantinya, Anda harus menggunakan URL asli dengan nomor Akun AWS ID Anda di <https://.signin.aws.amazon.com/console.<accountidnumber>>

```
Remove-IAMAccountAlias -AccountAlias mycompanyaws
```

- Untuk detail API, lihat [DeleteAccountAlias](#) di Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK for 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 [DeleteAccountAlias](#) di AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK for 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 [DeleteAccountAlias](#) di Referensi AWS SDK for Ruby API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan `DeleteAccountPasswordPolicy` dengan AWS SDK atau alat baris perintah

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 [DeleteAccountPasswordPolicy](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [DeleteAccountPasswordPolicy](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **DeleteGroup** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan `DeleteGroup`.

Contoh-contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan di dalam konteks. Anda dapat melihat tindakan ini dalam konteks pada contoh kode berikut:

- [Buat grup dan tambahkan pengguna](#)

.NET

AWS SDK for .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 group.
/// </summary>
/// <param name="groupName">The name of the IAM group to delete.</param>
/// <returns>A Boolean value indicating the success of the action.</returns>
public async Task<bool> DeleteGroupAsync(string groupName)
{
    var response = await _IAMService.DeleteGroupAsync(new DeleteGroupRequest
    { GroupName = groupName });
    return response.HttpStatusCode == HttpStatusCode.OK;
}
```

- Untuk detail API, lihat [DeleteGroup](#) di Referensi AWS SDK for .NET API.

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 [DeleteGroup](#) di 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 [DeleteGroup](#) di Referensi AWS SDK for JavaScript API.

PowerShell

Alat untuk PowerShell

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 [DeleteGroup](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **DeleteGroupPolicy** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan `DeleteGroupPolicy`.

Contoh-contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan di dalam konteks. Anda dapat melihat tindakan ini dalam konteks pada contoh kode berikut:

- [Buat grup dan tambahkan pengguna](#)

.NET

AWS SDK for .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 associated with an IAM group.
/// </summary>
/// <param name="groupName">The name of the IAM group associated with the
/// policy.</param>
/// <param name="policyName">The name of the policy to delete.</param>
/// <returns>A Boolean value indicating the success of the action.</returns>
public async Task<bool> DeleteGroupPolicyAsync(string groupName, string
policyName)
{
    var request = new DeleteGroupPolicyRequest()
    {
        GroupName = groupName,
        PolicyName = policyName,
    };

    var response = await _IAMService.DeleteGroupPolicyAsync(request);
    return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
}
```

- Untuk detail API, lihat [DeleteGroupPolicy](#) di Referensi AWS SDK for .NET API.

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 [DeleteGroupPolicy](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [DeleteGroupPolicy](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **DeleteInstanceProfile** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan `DeleteInstanceProfile`.

Contoh-contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan di dalam konteks. Anda dapat melihat tindakan ini dalam konteks pada contoh kode berikut:

- [Membangun dan mengelola layanan yang tangguh](#)

.NET

AWS SDK for .NET

Note

Ada lebih banyak tentang GitHub. Temukan contoh selengkapnya 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 [DeleteInstanceProfile](#) di Referensi AWS SDK for .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 [DeleteInstanceProfile](#) di 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 [DeleteInstanceProfile](#) di Referensi AWS SDK for JavaScript API.

PowerShell

Alat untuk PowerShell

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 [DeleteInstanceProfile](#) di Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK for 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 AutoScaler:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
```

```
"""

def __init__(
    self,
    resource_prefix,
    inst_type,
    ami_param,
    autoscaling_client,
    ec2_client,
    ssm_client,
    iam_client,
):
    """
    :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
    self.launch_template_name = f"{resource_prefix}-template"
    self.group_name = f"{resource_prefix}-group"
    self.instance_policy_name = f"{resource_prefix}-pol"
    self.instance_role_name = f"{resource_prefix}-role"
    self.instance_profile_name = f"{resource_prefix}-prof"
    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"
    self.key_pair_name = f"{resource_prefix}-key-pair"

def delete_instance_profile(self, profile_name, role_name):
    """
```

```

    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:
    if err.response["Error"]["Code"] == "NoSuchEntity":
        log.info(
            "Instance profile %s doesn't exist, nothing to do.",
profile_name
        )
    else:
        raise AutoScalerError(
            f"Couldn't delete instance profile {profile_name} or detach "
            f"policies and delete role {role_name}: {err}"
        )

```

- Untuk detail API, lihat [DeleteInstanceProfile](#) di AWS SDK for Python (Boto3) Referensi API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **DeleteLoginProfile** dengan AWS SDK atau alat baris perintah

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 [DeleteLoginProfile](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [DeleteLoginProfile](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **DeleteOpenIdConnectProvider** dengan AWS SDK atau alat baris perintah

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 [DeleteOpenIdConnectProvider](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [DeleteOpenIdConnectProvider](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **DeletePolicy** dengan AWS SDK atau alat baris perintah

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:

- [Buat pengguna dan ambil peran](#)
- [Buat pengguna read-only dan read-write](#)
- [Kelola kebijakan](#)

.NET

AWS SDK for .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 [DeletePolicy](#) di Referensi AWS SDK for .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 getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function iam_delete_policy"
        echo "Deletes an WS Identity and Access Management (IAM) policy"
        echo "  -n policy_arn -- The name of the IAM policy arn."
        echo ""
    }

    # Retrieve the calling parameters.
    while getopt "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 [DeletePolicy](#) di Referensi AWS CLI Perintah.

C++

SDK for 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 [DeletePolicy](#) di Referensi AWS SDK for 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 [DeletePolicy](#) di Referensi AWS CLI Perintah.

Go

SDK for Go V2

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
// 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(policyArn string) error {
    _, err := wrapper.IamClient.DeletePolicy(context.TODO(), &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 [DeletePolicy](#) di Referensi AWS SDK for Go API.

Java

SDK for 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 [DeletePolicy](#) di 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 [DeletePolicy](#) di Referensi AWS SDK for JavaScript API.

Kotlin

SDK for 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 { region = "AWS_GLOBAL" }.use { iamClient ->  
        iamClient.deletePolicy(request)  
        println("Successfully deleted $policyARNVal")  
    }  
}
```

- Untuk detail API, lihat [DeletePolicy](#) di AWS SDK untuk referensi API Kotlin.

PowerShell

Alat untuk PowerShell

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 [DeletePolicy](#) di Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK for 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 [DeletePolicy](#) di 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 [DeletePolicy](#) referensi AWS SDK for Rust API.

Swift

SDK for Swift

Note

Ini adalah dokumentasi prarilis untuk SDK dalam rilis pratinjau. Dokumentasi ini dapat berubah.

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
public func deletePolicy(policy: IAMClientTypes.Policy) async throws {
    let input = DeletePolicyInput(
        policyArn: policy.arn
    )
    do {
        _ = try await iamClient.deletePolicy(input: input)
    } catch {
        throw error
    }
}
```

- Untuk detail API, lihat referensi [DeletePolicy AWSSDK](#) untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **DeletePolicyVersion** dengan AWS SDK atau alat baris perintah

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 [DeletePolicyVersion](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [DeletePolicyVersion](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **DeleteRole** dengan AWS SDK atau alat baris perintah


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:

- [Buat pengguna dan ambil peran](#)
- [Kelola peran](#)

.NET

AWS SDK for .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 [DeleteRole](#) di Referensi AWS SDK for .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 getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function iam_delete_role"
        echo "Deletes an WS Identity and Access Management (IAM) role"
        echo "  -n role_name -- The name of the IAM role."
        echo ""
    }

    # Retrieve the calling parameters.
    while getopt "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 [DeleteRole](#) di 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 [DeleteRole](#) di Referensi AWS CLI Perintah.

Go

SDK for Go V2

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
// 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(roleName string) error {
    _, err := wrapper.IamClient.DeleteRole(context.TODO(), &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 [DeleteRole](#) di Referensi AWS SDK for 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 [DeleteRole](#) di Referensi AWS SDK for JavaScript API.

PowerShell

Alat untuk PowerShell

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 [DeleteRole](#) di Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK for 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 [DeleteRole](#) di AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK for 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)
  begin
    # 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
  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
end
```

- Untuk detail API, lihat [DeleteRole](#) di Referensi AWS SDK for 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 [DeleteRole](#) referensi AWS SDK for Rust API.

Swift

SDK for Swift

Note

Ini adalah dokumentasi prarilis untuk SDK dalam rilis pratinjau. Dokumentasi ini dapat berubah.

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
public func deleteRole(role: IAMClientTypes.Role) async throws {
    let input = DeleteRoleInput(
        roleName: role.roleName
    )
    do {
        _ = try await iamClient.deleteRole(input: input)
    } catch {
        throw error
    }
}
```

- Untuk detail API, lihat referensi [DeleteRole AWSSDK](#) untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **DeleteRolePermissionsBoundary** dengan AWS SDK atau alat baris perintah

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 [DeleteRolePermissionsBoundary](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini menunjukkan cara menghapus batas izin yang dilampirkan ke peran IAM.

```
Remove-IAMRolePermissionsBoundary -RoleName MyRoleName
```

- Untuk detail API, lihat [DeleteRolePermissionsBoundary](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **DeleteRolePolicy** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan `DeleteRolePolicy`.

.NET

AWS SDK for .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 [DeleteRolePolicy](#) di Referensi AWS SDK for .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 [DeleteRolePolicy](#) di 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 [DeleteRolePolicy](#) di Referensi AWS SDK for JavaScript API.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini menghapus kebijakan inline **S3AccessPolicy** yang disematkan dalam peran IAM. **S3BackupRole**

```
Remove-IAMRolePolicy -PolicyName S3AccessPolicy -RoleName S3BackupRole
```

- Untuk detail API, lihat [DeleteRolePolicy](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **DeleteSAMLProvider** dengan AWS SDK atau alat baris perintah

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 for JavaScript

PowerShell

Alat untuk PowerShell

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 Cmdlet](#). AWS Tools for PowerShell


Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **DeleteServerCertificate** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan `DeleteServerCertificate`.

C++

SDK for 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 [DeleteServerCertificate](#) di Referensi AWS SDK for 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 [DeleteServerCertificate](#) di 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 for JavaScript](#).
- Untuk detail API, lihat [DeleteServerCertificate](#) di Referensi AWS SDK for 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 for JavaScript](#).
- Untuk detail API, lihat [DeleteServerCertificate](#) di Referensi AWS SDK for JavaScript API.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini menghapus sertifikat server bernama **MyServerCert**.

```
Remove-IAMServerCertificate -ServerCertificateName MyServerCert
```

- Untuk detail API, lihat [DeleteServerCertificate](#) di Referensi AWS Tools for PowerShell Cmdlet.

Ruby

SDK for 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
  end
end
```

```
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
end
```

- Untuk detail API, lihat [DeleteServerCertificate](#) di Referensi AWS SDK for Ruby API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **DeleteServiceLinkedRole** dengan AWS SDK atau alat baris perintah

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 [DeleteServiceLinkedRole](#) di Referensi AWS CLI Perintah.

Go

SDK for Go V2

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
// 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(roleName string) error {
    _, err := wrapper.IamClient.DeleteServiceLinkedRole(context.TODO(),
        &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 [DeleteServiceLinkedRole](#) di Referensi AWS SDK for 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 [DeleteServiceLinkedRole](#) di Referensi AWS SDK for JavaScript API.

PowerShell

Alat untuk PowerShell

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 [DeleteServiceLinkedRole](#) di Referensi AWS Tools for PowerShell Cmdlet.

Ruby

SDK for 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)
  unless e.code == "NoSuchEntity"
    @logger.error("Couldn't delete #{role_name}. Here's why:")
    @logger.error("\t#{e.code}: #{e.message}")
    raise
  end
end
```

- Untuk detail API, lihat [DeleteServiceLinkedRole](#) di Referensi AWS SDK for 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 [DeleteServiceLinkedRole](#) referensi AWS SDK for Rust API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **DeleteSigningCertificate** dengan AWS SDK atau alat baris perintah

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 TA7SMP42TDN5Z260BPJE7EXAMPLE
```

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 [DeleteSigningCertificate](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [DeleteSigningCertificate](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **DeleteUser** dengan AWS SDK atau alat baris perintah

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:

- [Buat grup dan tambahkan pengguna](#)
- [Buat pengguna dan ambil peran](#)
- [Buat pengguna read-only dan read-write](#)

.NET

AWS SDK for .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 [DeleteUser](#) di Referensi AWS SDK for .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 getopt command in a function.  
  
    # bashsupport disable=BP5008  
    function usage() {  
        echo "function iam_delete_user"  
        echo "Deletes an WS 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 getopt "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."  
    fi  
}
```

```
    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 [DeleteUser](#) di Referensi AWS CLI Perintah.

C++

SDK for 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 [DeleteUser](#) di Referensi AWS SDK for 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 [DeleteUser](#) di Referensi AWS CLI Perintah.

Go

SDK for Go V2

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
// 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(userName string) error {
    _, err := wrapper.IamClient.DeleteUser(context.TODO(), &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 [DeleteUser](#) di Referensi AWS SDK for Go API.

Java

SDK for 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 [DeleteUser](#) di 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 for JavaScript](#).
- Untuk detail API, lihat [DeleteUser](#) di Referensi AWS SDK for 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 for JavaScript](#).
- Untuk detail API, lihat [DeleteUser](#) di Referensi AWS SDK for JavaScript API.

Kotlin

SDK for 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 { region = "AWS_GLOBAL" }.use { iamClient ->
    iamClient.deleteUser(request)
    println("Successfully deleted user $userNameVal")
}
}
```

- Untuk detail API, lihat [DeleteUser](#) di AWS SDK untuk referensi API Kotlin.

PowerShell

Alat untuk PowerShell

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 [DeleteUser](#) di Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK for 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 [DeleteUser](#) di AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK for 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
```

```
end
```

- Untuk detail API, lihat [DeleteUser](#) di Referensi AWS SDK for 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 [DeleteUser](#) referensi AWS SDK for Rust API.

Swift

SDK for Swift

Note

Ini adalah dokumentasi prarilis untuk SDK dalam rilis pratinjau. Dokumentasi ini dapat berubah.

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
public func deleteUser(user: IAMClientTypes.User) async throws {
    let input = DeleteUserInput(
        userName: user.userName
    )
    do {
        _ = try await iamClient.deleteUser(input: input)
    } catch {
        throw error
    }
}
```

- Untuk detail API, lihat referensi [DeleteUser AWSSDK](#) untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan `DeleteUserPermissionsBoundary` dengan AWS SDK atau alat baris perintah

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 [DeleteUserPermissionsBoundary](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini menunjukkan cara menghapus batas izin yang dilampirkan ke pengguna IAM.

```
Remove-IAMUserPermissionsBoundary -UserName joe
```

- Untuk detail API, lihat [DeleteUserPermissionsBoundary](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **DeleteUserPolicy** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan `DeleteUserPolicy`.

Contoh-contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan di dalam konteks. Anda dapat melihat tindakan ini dalam konteks pada contoh kode berikut:

- [Buat pengguna dan ambil peran](#)

.NET

AWS SDK for .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 [DeleteUserPolicy](#) di Referensi AWS SDK for .NET API.

CLI

AWS CLI

Untuk menghapus kebijakan dari pengguna IAM

`delete-user-policy` Perintah berikut menghapus kebijakan yang ditentukan dari pengguna IAM bernama Bob.

```
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 [DeleteUserPolicy](#) di Referensi AWS CLI Perintah.

Go

SDK for Go V2

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
// 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(userName string, policyName string)
error {
    _, err := wrapper.IamClient.DeleteUserPolicy(context.TODO(),
    &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 [DeleteUserPolicy](#) di Referensi AWS SDK for Go API.

PowerShell

Alat untuk PowerShell

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 [DeleteUserPolicy](#) di Referensi AWS Tools for PowerShell Cmdlet.

Ruby

SDK for 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 [DeleteUserPolicy](#) di Referensi AWS SDK for 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 [DeleteUserPolicy](#) referensi AWS SDK for Rust API.

Swift

SDK for Swift

Note

Ini adalah dokumentasi prarilis untuk SDK dalam rilis pratinjau. Dokumentasi ini dapat berubah.

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
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 {
  throw error
}
}
```

- Untuk detail API, lihat referensi [DeleteUserPolicy AWS SDK](#) untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **DeleteVirtualMfaDevice** dengan AWS SDK atau alat baris perintah

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 [DeleteVirtualMfaDevice](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [DeleteVirtualMfaDevice](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **DetachGroupPolicy** dengan AWS SDK atau alat baris perintah

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 [DetachGroupPolicy](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [DetachGroupPolicy](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **DetachRolePolicy** dengan AWS SDK atau alat baris perintah

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:

- [Buat pengguna dan ambil peran](#)
- [Kelola peran](#)

.NET

AWS SDK for .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 [DetachRolePolicy](#) di Referensi AWS SDK for .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 role.
#
# 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 getopt 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
```

```
fi

echo "$response"

return 0
}
```

- Untuk detail API, lihat [DetachRolePolicy](#) di Referensi AWS CLI Perintah.

C++

SDK for 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 [DetachRolePolicy](#) di Referensi AWS SDK for 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 [DetachRolePolicy](#) di Referensi AWS CLI Perintah.

Go

SDK for Go V2

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
// 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(roleName string, policyArn string)
error {
    _, err := wrapper.IamClient.DetachRolePolicy(context.TODO(),
&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 [DetachRolePolicy](#) di Referensi AWS SDK for Go API.

Java

SDK for 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 [DetachRolePolicy](#) di 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 for JavaScript](#).
- Untuk detail API, lihat [DetachRolePolicy](#) di Referensi AWS SDK for JavaScript API.

SDK untuk JavaScript (v2)

Note

Ada lebih banyak tentang GitHub. Temukan contoh selengkapnya 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 for JavaScript](#).
- Untuk detail API, lihat [DetachRolePolicy](#) di Referensi AWS SDK for JavaScript API.

Kotlin

SDK for 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 { region = "AWS_GLOBAL" }.use { iamClient ->  
        iamClient.detachRolePolicy(request)  
        println("Successfully detached policy $policyArnVal from role  
$roleNameVal")  
    }  
}
```

- Untuk detail API, lihat [DetachRolePolicy](#) di AWS SDK untuk referensi API Kotlin.

PowerShell

Alat untuk PowerShell

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 [DetachRolePolicy](#) di Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK for 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 [DetachRolePolicy](#) di AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK for 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 [DetachRolePolicy](#) di Referensi AWS SDK for 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 [DetachRolePolicy](#) referensi AWS SDK for Rust API.

Swift

SDK for Swift

Note

Ini adalah dokumentasi prarilis untuk SDK dalam rilis pratinjau. Dokumentasi ini dapat berubah.

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
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 {
        throw error
    }
}
```

- Untuk detail API, lihat referensi [DetachRolePolicy AWSSDK](#) untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **DetachUserPolicy** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan `DetachUserPolicy`.

Contoh-contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan di dalam konteks. Anda dapat melihat tindakan ini dalam konteks pada 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 [DetachUserPolicy](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [DetachUserPolicy](#) di Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK for 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 [DetachUserPolicy](#) di AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK for 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 [DetachUserPolicy](#) di Referensi AWS SDK for 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 [DetachUserPolicy](#) referensi AWS SDK for Rust API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **EnableMfaDevice** dengan AWS SDK atau alat baris perintah

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 [EnableMfaDevice](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [EnableMfaDevice](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **GenerateCredentialReport** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan `GenerateCredentialReport`.

Contoh-contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan di dalam konteks. Anda dapat melihat tindakan ini dalam konteks pada 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 [GenerateCredentialReport](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [GenerateCredentialReport](#) di Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK for 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 [GenerateCredentialReport](#) di AWS SDK for Python (Boto3) Referensi API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **GenerateServiceLastAccessedDetails** dengan AWS SDK atau alat baris perintah

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 kustom 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 [GenerateServiceLastAccessedDetails](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini setara dengan cmdlet API. `GenerateServiceLastAccessedDetails` Ini menyediakan id pekerjaan yang dapat digunakan di `Get-IAM` dan `ServiceLastAccessedDetail` `Get-IAM ServiceLastAccessedDetailWithEntity`

```
Request-IAMServiceLastAccessedDetail -Arn arn:aws:iam::123456789012:user/TestUser
```

- Untuk detail API, lihat [GenerateServiceLastAccessedDetails](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **GetAccessKeyLastUsed** dengan AWS SDK atau alat baris perintah


Contoh kode berikut menunjukkan cara menggunakan `GetAccessKeyLastUsed`.

Contoh-contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan di dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- [Kelola kunci akses](#)

C++

SDK for 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 [GetAccessKeyLastUsed](#) di Referensi AWS SDK for 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 [GetAccessKeyLastUsed](#) di 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 for JavaScript](#).
- Untuk detail API, lihat [GetAccessKeyLastUsed](#) di Referensi AWS SDK for 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 for JavaScript](#).
- Untuk detail API, lihat [GetAccessKeyLastUsed](#) di Referensi AWS SDK for JavaScript API.

PowerShell

Alat untuk PowerShell

Contoh 1: Mengembalikan nama pengguna yang dimiliki dan informasi penggunaan terakhir untuk kunci akses yang disediakan.

```
Get-IAMAccessKeyLastUsed -AccessKeyId ABCDEXAMPLE
```

- Untuk detail API, lihat [GetAccessKeyLastUsed](#) di Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK for 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 [GetAccessKeyLastUsed](#) di AWS SDK for Python (Boto3) Referensi API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **GetAccountAuthorizationDetails** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan `GetAccountAuthorizationDetails`.

Contoh-contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan di dalam konteks. Anda dapat melihat tindakan ini dalam konteks pada contoh kode berikut:

- [Kelola akun Anda](#)

CLI

AWS CLI

Untuk mencantumkan AWS akun, pengguna, grup, peran, dan kebijakan IAM

`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": "Stmnt1381777017000",
              "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:::example-bucket",
                    "arn:aws:s3:::example-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": "EXAMPLEkakov9BCuUNFDtxWSyetzYwEx2ADc8dnzfvERF5S6YMvXKx41t6gCl/
eeaCX3Jo94/bKqezEAg8TEVS99EKFLxm3jtbpl25FDWEXAMPLE",
"IsTruncated": true
}

```

Untuk informasi selengkapnya, lihat [pedoman audit AWS keamanan](#) di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat [GetAccountAuthorizationDetails](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [GetAccountAuthorizationDetails](#) di Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK for 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 [GetAccountAuthorizationDetails](#) di AWS SDK for Python (Boto3) Referensi API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **GetAccountPasswordPolicy** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan `GetAccountPasswordPolicy`.

.NET

AWS SDK for .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 [GetAccountPasswordPolicy](#) di Referensi AWS SDK for .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 [GetAccountPasswordPolicy](#) di Referensi AWS CLI Perintah.

Go

SDK for Go V2

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
// 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() (*types.PasswordPolicy,
error) {
    var pwPolicy *types.PasswordPolicy
    result, err := wrapper.IamClient.GetAccountPasswordPolicy(context.TODO(),
    &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 [GetAccountPasswordPolicy](#) di Referensi AWS SDK for 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 [GetAccountPasswordPolicy](#) di Referensi AWS SDK for JavaScript API.

PHP

SDK for 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 [GetAccountPasswordPolicy](#) di Referensi AWS SDK for PHP API.

PowerShell

Alat untuk PowerShell

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 [GetAccountPasswordPolicy](#) di Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK for 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 [GetAccountPasswordPolicy](#) di AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK for 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
  begin
    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
end
end
```

- Untuk detail API, lihat [GetAccountPasswordPolicy](#) di Referensi AWS SDK for 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 [GetAccountPasswordPolicy](#) referensi AWS SDK for Rust API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **GetAccountSummary** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan `GetAccountSummary`.

Contoh-contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan di dalam konteks. Anda dapat melihat tindakan ini dalam konteks pada 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 [GetAccountSummary](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [GetAccountSummary](#) di Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK for 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 [GetAccountSummary](#) di AWS SDK for Python (Boto3) Referensi API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **GetContextKeysForCustomPolicy** dengan AWS SDK atau alat baris perintah

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 [GetContextKeysForCustomPolicy](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [GetContextKeysForCustomPolicy](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan `GetContextKeysForPrincipalPolicy` dengan AWS SDK atau alat baris perintah

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 [GetContextKeysForPrincipalPolicy](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengambil semua kunci konteks yang ada dalam json kebijakan yang disediakan dan kebijakan yang dilampirkan ke entitas IAM (pengguna/peran, 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 [GetContextKeysForPrincipalPolicy](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **GetCredentialReport** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan `GetCredentialReport`.

Contoh-contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan di dalam konteks. Anda dapat melihat tindakan ini dalam konteks pada 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 [GetCredentialReport](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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,password_last_changed,pa
root_account,arn:aws:iam::123456789012:root,2014-10-15T16:31:25+00:00,not_supported,2015-04-15T16:31:25+00:00,
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+00:00,2014-10-16T16:06:00,2014-10-16T16:06:00,
A,false,true,2014-12-03T18:53:41+00:00,true,2015-03-25T20:38:14+00:00,false,N/A
A,false,N/A
Bill,arn:aws:iam::123456789012:user/Bill,2015-04-15T18:27:44+00:00,false,N/A,N/A
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 [GetCredentialReport](#) di Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK for 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 [GetCredentialReport](#) di AWS SDK for Python (Boto3) Referensi API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **GetGroup** dengan AWS SDK atau alat baris perintah

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": "AIDGPMS9R04H3FEXAMPLE",
    "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 [GetGroup](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengembalikan detail tentang grup **IAMTesters**, termasuk kumpulan semua pengguna IAM yang termasuk dalam grup.

```
$results = Get-IAMGroup -GroupName "Testers"
$results
```

Output:

Group	IsTruncated	Marker
Users		
-----	-----	-----
Amazon.IdentityManagement.Model.Group {Theresa, David}	False	

```
$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    : 40SVDDJJTF4XEEXAMPLE2
```

```

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 [GetGroup](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **GetGroupPolicy** dengan AWS SDK atau alat baris perintah

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 [GetGroupPolicy](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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
-----
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",

```

```
    "NotAction": "iam:*",
    "Resource": "*"
  }
]
}
```

- Untuk detail API, lihat [GetGroupPolicy](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **GetInstanceProfile** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan `GetInstanceProfile`.

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": {
    "InstanceId": "AID2MAB8DPLSRHEXAMPLE",
    "Roles": [
      {
        "AssumeRolePolicyDocument": "<URL-encoded-JSON>",
        "RoleId": "AIDGPMS9R04H3FEXAMPLE",
        "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 [GetInstanceProfile](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [GetInstanceProfile](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **GetLoginProfile** dengan AWS SDK atau alat baris perintah

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 bernama Bob.

```
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 [GetLoginProfile](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [GetLoginProfile](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **GetOpenIdConnectProvider** dengan AWS SDK atau alat baris perintah

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 [GetOpenIdConnectProvider](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 Url	CreateDate	ThumbprintList
----- ---	-----	-----
{MyOIDCApp} {12345abcdefghijkl67890lmnopqrst98765uvwxyz}	2/3/2015 3:00:30 PM	oidc.example.com

- Untuk detail API, lihat [GetOpenIdConnectProvider](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **GetPolicy** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan `GetPolicy`.

Contoh-contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan di dalam konteks. Anda dapat melihat tindakan ini dalam konteks pada contoh kode berikut:

- [Bekerja dengan API Pembuat Kebijakan IAM](#)

.NET

AWS SDK for .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 [GetPolicy](#) di Referensi AWS SDK for .NET API.

C++

SDK for 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 [GetPolicy](#) di Referensi AWS SDK for 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 [GetPolicy](#) di Referensi AWS CLI Perintah.

Go

SDK for Go V2

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
// 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(policyArn string) (*types.Policy, error) {
    var policy *types.Policy
    result, err := wrapper.IamClient.GetPolicy(context.TODO(), &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 [GetPolicy](#) di Referensi AWS SDK for 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 for JavaScript](#).
- Untuk detail API, lihat [GetPolicy](#) di Referensi AWS SDK for 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 for JavaScript](#).
- Untuk detail API, lihat [GetPolicy](#) di Referensi AWS SDK for JavaScript API.

Kotlin

SDK for 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 { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.getPolicy(request)
        println("Successfully retrieved policy ${response.policy?.policyName}")
    }
}
```

- Untuk detail API, lihat [GetPolicy](#) di AWS SDK untuk referensi API Kotlin.

PHP

SDK for 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 [GetPolicy](#) di Referensi AWS SDK for PHP API.

PowerShell

Alat untuk PowerShell

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      : Z27SI6FQMGNO2EXAMPLE1
PolicyName    : MySamplePolicy
UpdateDate   : 2/6/2015 10:40:08 AM
```

- Untuk detail API, lihat [GetPolicy](#) di Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK for 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 [GetPolicy](#) di AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK for 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 [GetPolicy](#) di Referensi AWS SDK for Ruby API.

Swift

SDK for Swift

Note

Ini adalah dokumentasi prarilis untuk SDK dalam rilis pratinjau. Dokumentasi ini dapat berubah.

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
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 {
        throw error
    }
}
```

- Untuk detail API, lihat referensi [GetPolicy AWSSDK](#) untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **GetPolicyVersion** dengan AWS SDK atau alat baris perintah

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-nya. `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 [GetPolicyVersion](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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": "*",
    "Resource": "*"
  }
}
```

- Untuk detail API, lihat [GetPolicyVersion](#) di Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK for 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 [GetPolicyVersion](#) di AWS SDK for Python (Boto3) Referensi API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **GetRole** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan `GetRole`.

.NET

AWS SDK for .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 [GetRole](#) di Referensi AWS SDK for .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 tersebut. 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 [GetRole](#) di Referensi AWS CLI Perintah.

Go

SDK for Go V2

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
// 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(roleName string) (*types.Role, error) {
    var role *types.Role
    result, err := wrapper.IamClient.GetRole(context.TODO(),
        &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 [GetRole](#) di Referensi AWS SDK for 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 perannya.

```
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 [GetRole](#) di Referensi AWS SDK for JavaScript API.

PHP

SDK for 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 [GetRole](#) di Referensi AWS SDK for PHP API.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengembalikan rincian **lambda_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 [GetRole](#) di Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK for 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 [GetRole](#) di AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK for 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 [GetRole](#) di Referensi AWS SDK for 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 [GetRole](#) referensi AWS SDK for Rust API.

Swift

SDK for Swift

Note

Ini adalah dokumentasi prarilis untuk SDK dalam rilis pratinjau. Dokumentasi ini dapat berubah.

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
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 {
        throw error
    }
}
```

- Untuk detail API, lihat referensi [GetRole AWSSDK](#) untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **GetRolePolicy** dengan AWS SDK atau alat baris perintah

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 [GetRolePolicy](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengembalikan dokumen kebijakan izin untuk kebijakan bernama **oneClick_lambda_exec_role_policy** yang disematkan dalam peran IAM.

lambda_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
<pre> UserName ----- ----- %7B%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C%... oneClick_lambda_exec_role_policy lambda_exec_role </pre>	

```
[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:*:*:*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",

```

```

        "s3:PutObject"
      ],
      "Resource": [
        "arn:aws:s3:::*"
      ]
    }
  ]
}

```

- Untuk detail API, lihat [GetRolePolicy](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **GetSamlProvider** dengan AWS SDK atau alat baris perintah

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 [GetSamlProvider](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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	ValidUntil	SAMLMetadataDocument
-----	-----	-----
12/23/2014 12:16:55 PM	12/23/2114 12:16:54 PM	<EntityDescriptor ID="_12345678-1234-5678-9012-example1...

- Untuk detail API, lihat [GetSamlProvider](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **GetServerCertificate** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan `GetServerCertificate`.

C++

SDK for 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 [GetServerCertificate](#) di Referensi AWS SDK for 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-----
MIICiTCCAfICCQD6m7oRw0uX0jANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
VVMxCzAJBgNVBAGTAldBMRAwDgYDVQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6
b24xFDASBgNVBAsTC0lBTSBDb25zb2x1MRIwEAYDVQQDEw1UZXRhbnQ21sYWMxHzAd
BgkqhkiG9w0BCQEWEG5vb251QGFTYXpva251b20wHhcNMTEwNDIwMTIwMjIwMjIw
MTIwNDIwMTIwMjIwMjIwMjIwMjIwMjIwMjIwMjIwMjIwMjIwMjIwMjIwMjIwMjIw
VQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z

```

```

b2x1MRIwEAYDVQQDEw1UZlXN0Q21sYWMxHzAdBgkqhkiG9w0BCQEWEG5vb251QGFt
YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEI03IyNoH/f0wYK8m9T
rDHudUZg3qX4waLG5M43q7Wgc/MbQITx0USQv7c7ugFFDzQGBzZswY6786m86gpE
Ibb30hjZnzcVQAaRHhd1QWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwax1Aoo7TJHidbtS4J5iNmZgXL0Fkb
FFBjvSfpJI1J00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp3780D8uTs7fLvJx79LjSTb
NYiytVbZPQUQ5Yaxu2jXnimvrszlaEXAMPLE=-----END CERTIFICATE-----",
"CertificateChain": "-----BEGIN CERTIFICATE-----\nMIICiTCCAfICQD6md
7oRw0uX0jANBgkqhkiG9w0BAQQUFADCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAGT
AldBMRAdGyYDVQOHEwdTZWF0drGx1MQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBA
sTC01BTSBDb25zb2x1MRIwEAYDVQDEw1UZlXN0Q21sYWMxHzAdBgkqhkiG9w0BCQ
jb20wHhcNMTEwNDI1MjA0NTIxWhcNMTEwNDI0MjA0NTIxWjCBiDELMAkGA1UEBh
MCVVMxCzAJBgNVBAGTAldBMRAwDgsYDVQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBb
WF6b24xFDASBgNVBAstC01BTSBDb2d5zb2x1MRIwEAYDVQOHEwdTZWF0dGx1MQ8w
DQYDVQQKEwZBbWF6b24xFDASBgNVBAstC01BTSBDb2d5zb2x1MRIwEAYDVQOHEw
1UZlXN0Q21sYWMxHzAdBgkqhkiG9w0BCQEWEG5vb251QGFtYXpvbi5jb20wgZ8w
DQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIgWJ21uUSfwfEvySWt
C2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEI03IyNoH/f0wYK8mh9TrDHudUZg3qX4
waLG5M43q7Wgc/MbQITx0USQv7c7ugFFDzQGBzZswY6786m86gjpEIbb30hjZnzc
VQAaRHhd1QWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCku4nUhVVxYUnt
neD9+h8Mg9q6q+auNKyExzyLwax1Aoo7TJHidbtS4J5iNmZgXL0F1kbFFBjvSfp
JI1J00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp3780D8uTs7fLvJx79LjS;TbNYiytVb
ZPQUQ5Yaxu2jXnimv3rrszlaEWEG5vb251QGFtsYXpvbiEXAMPLE=\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 [GetServerCertificated](#) di 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 for JavaScript](#).
- Untuk detail API, lihat [GetServerCertificate](#) di Referensi AWS SDK for 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 for JavaScript](#).
- Untuk detail API, lihat [GetServerCertificate](#) di Referensi AWS SDK for JavaScript API.

PowerShell

Alat untuk PowerShell

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-----

MIICiTCCAfICCQD6m7oRw0uX0jANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC

VVMxCzAJBgNVBAGTAldBMRAwDgYDVQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6

b24xFDASBgNVBAstC01BTSBDb25zb2x1MRIwEAYDVQQDEw1UZXR0Q21sYWMxHzAd

BgkqhkiG9w0BCQEWEG5vb251QGftYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN

MTIwNDI1MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAGTAldBMRAwDgYD

VQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAstC01BTSBDb25z

b2x1MRIwEAYDVQQDEw1UZXR0Q21sYWMxHzAdBgkqhkiG9w0BCQEWEG5vb251QGft
```

```

YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
      21uUSfwfEvySWtC2XADZ4nB
+BLyGVik60CpiwsZ3G93vUEI03IyNoH/f0wYK8m9T
      rDHudUZg3qX4waLG5M43q7Wgc/
MbQITx0USQv7c7ugFFDzQGBzZswY6786m86gpE

Ibb30hjZnzcVQAaRHhd1QWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
      nUhVVxYUntneD9+h8Mg9q6q
+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb

FFBjvSfpJl1J00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp3780D8uTs7fLvJx79LjSTb
      NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
      -----END CERTIFICATE-----
CertificateChain      :
ServerCertificateMetadata :
  Amazon.IdentityManagement.Model.ServerCertificateMetadata

```

```
$result.ServerCertificateMetadata
```

Output:

```

Arn          : arn:aws:iam::123456789012:server-certificate/0rg1/0rg2/
MyServerCertificate
Expiration   : 1/14/2018 9:52:36 AM
Path        : /0rg1/0rg2/
ServerCertificateId : ASCAJIFEXAMPLE17HQZYW
ServerCertificateName : MyServerCertificate
UploadDate   : 4/21/2015 11:14:16 AM

```

- Untuk detail API, lihat [GetServerCertificate](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **GetServiceLastAccessedDetails** dengan AWS SDK atau alat baris perintah

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 [GetServiceLastAccessedDetails](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [GetServiceLastAccessedDetails](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **GetServiceLastAccessedDetailsWithEntities** dengan AWS SDK atau alat baris perintah

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": "AIDAI02XMPLNQEXAMPLE",
        "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 [GetServiceLastAccessedDetailsWithEntities](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [GetServiceLastAccessedDetailsWithEntities](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan `GetServiceLinkedRoleDeletionStatus` dengan AWS SDK atau alat baris perintah

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 [GetServiceLinkedRoleDeletionStatus](#) di 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 [GetServiceLinkedRoleDeletionStatus](#) di Referensi AWS SDK for JavaScript API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **GetUser** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan `GetUser`.

.NET

AWS SDK for .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 [GetUser](#) di Referensi AWS SDK for .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 [GetUser](#) di 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 [GetUser](#) di Referensi AWS CLI Perintah.

Go

SDK for Go V2

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
// 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(userName string) (*types.User, error) {
    var user *types.User
    result, err := wrapper.IamClient.GetUser(context.TODO(), &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 [GetUser](#) di Referensi AWS SDK for Go API.

PowerShell

Alat untuk PowerShell

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 [GetUser](#) di Referensi AWS Tools for PowerShell Cmdlet.

Ruby

SDK for 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 [GetUser](#) di Referensi AWS SDK for Ruby API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **GetUserPolicy** dengan AWS SDK atau alat baris perintah

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 bernama `Bob`.

```
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 [GetUserPolicy](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengambil rincian kebijakan inline bernama **Dauids_IAM_Admin_Policy** yang disematkan dalam nama pengguna IAM. **David** Dokumen kebijakan adalah URL yang dikodekan.

```
$results = Get-IAMUserPolicy -PolicyName Dauids_IAM_Admin_Policy -UserName David
$results
```

Output:

```
PolicyDocument                                     PolicyName
-----
-----
%7B%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C%...   Dauids_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:*"
      ],
      "Resource": [
        "*"
      ]
    }
  ]
}
```

```
]
}
```

- Untuk detail API, lihat [GetUserPolicy](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **ListAccessKeys** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan `ListAccessKeys`.

Contoh-contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan di dalam konteks. Anda dapat melihat tindakan ini dalam konteks pada 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 getopt command in a function.
    # Retrieve the calling parameters.
    while getopt "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
    fi
}
```

```
    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 [ListAccessKeys](#) di Referensi AWS CLI Perintah.

C++

SDK for 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 [ListAccessKeys](#) di Referensi AWS SDK for 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 [ListAccessKeys](#) di Referensi AWS CLI Perintah.

Go

SDK for Go V2

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
// 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(userName string)
([]types.AccessKeyMetadata, error) {
    var keys []types.AccessKeyMetadata
    result, err := wrapper.IamClient.ListAccessKeys(context.TODO(),
&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 [ListAccessKeys](#) di Referensi AWS SDK for Go API.

Java

SDK for 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 [ListAccessKeys](#) di 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 for JavaScript](#).
- Untuk detail API, lihat [ListAccessKeys](#) di Referensi AWS SDK for 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 for JavaScript](#).
- Untuk detail API, lihat [ListAccessKeys](#) di Referensi AWS SDK for JavaScript API.

Kotlin

SDK for 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 { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listAccessKeys(request)
        response.accessKeyMetadata?.forEach { md ->
            println("Retrieved access key ${md.accessKeyId}")
        }
    }
}
```

```

    }
  }
}

```

- Untuk detail API, lihat [ListAccessKeys](#) di AWS SDK untuk referensi API Kotlin.

PowerShell

Alat untuk PowerShell

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 [ListAccessKeys](#) di Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK for 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 [ListAccessKeys](#) di AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK for 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 => e
  @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 => e
  @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 [ListAccessKeys](#) di Referensi AWS SDK for Ruby API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **ListAccountAliases** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan `ListAccountAliases`.

Contoh-contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan di dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- [Kelola akun Anda](#)

C++

SDK for 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 [ListAccountAliases](#) di Referensi AWS SDK for 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 [ListAccountAliases](#) di Referensi AWS CLI Perintah.

Java

SDK for 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 [ListAccountAliases](#) di 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 for JavaScript](#).
- Untuk detail API, lihat [ListAccountAliases](#) di Referensi AWS SDK for 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 for JavaScript](#).
- Untuk detail API, lihat [ListAccountAliases](#) di Referensi AWS SDK for JavaScript API.

Kotlin

SDK for 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 { region = "AWS_GLOBAL" }.use { iamClient ->  
        val response = iamClient.listAccountAliases(ListAccountAliasesRequest {})  
        response.accountAliases?.forEach { alias ->  
            println("Retrieved account alias $alias")  
        }  
    }  
}
```

- Untuk detail API, lihat [ListAccountAliases](#) di AWS SDK untuk referensi API Kotlin.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini mengembalikan alias akun untuk. Akun AWS

```
Get-IAMAccountAlias
```

Output:

```
ExampleCo
```

- Untuk detail API, lihat [ListAccountAliases](#) di Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK for 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 [ListAccountAliases](#) di AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK for 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
  end
end
```

```
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 [ListAccountAliases](#) di Referensi AWS SDK for Ruby API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **ListAttachedGroupPolicies** dengan AWS SDK atau alat baris perintah

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 [ListAttachedGroupPolicies](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [ListAttachedGroupPolicies](#) di Referensi AWS Tools for PowerShell Cmdlet.


Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **ListAttachedRolePolicies** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan `ListAttachedRolePolicies`.

.NET

AWS SDK for .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 [ListAttachedRolePolicies](#) di Referensi AWS SDK for .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 [ListAttachedRolePolicies](#) di Referensi AWS CLI Perintah.

Go

SDK for Go V2

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
// 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(roleName string)
([]types.AttachedPolicy, error) {
    var policies []types.AttachedPolicy
    result, err := wrapper.IamClient.ListAttachedRolePolicies(context.TODO(),
    &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 [ListAttachedRolePolicies](#) di Referensi AWS SDK for 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 [ListAttachedRolePolicies](#) di Referensi AWS SDK for JavaScript API.

PHP

SDK for 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 [ListAttachedRolePolicies](#) di Referensi AWS SDK for PHP API.

PowerShell

Alat untuk PowerShell

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 [ListAttachedRolePolicies](#) di Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK for 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 [ListAttachedRolePolicies](#) di AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK for 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 [ListAttachedRolePolicies](#) di Referensi AWS SDK for 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 [ListAttachedRolePolicies](#) referensi AWS SDK for Rust API.

Swift

SDK for Swift

Note

Ini adalah dokumentasi prarilis untuk SDK dalam rilis pratinjau. Dokumentasi ini dapat berubah.

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
/// 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] = []
    var marker: String? = nil
    var isTruncated: Bool
```

```
repeat {
    let input = ListAttachedRolePoliciesInput(
        marker: marker,
        roleName: role
    )
    let output = try await client.listAttachedRolePolicies(input: input)

    guard let attachedPolicies = output.attachedPolicies else {
        return policyList
    }

    for attachedPolicy in attachedPolicies {
        policyList.append(attachedPolicy)
    }
    marker = output.marker
    isTruncated = output.isTruncated
} while isTruncated == true
return policyList
}
```

- Untuk detail API, lihat referensi [ListAttachedRolePolicies AWSSDK](#) untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **ListAttachedUserPolicies** dengan AWS SDK atau alat baris perintah

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 [ListAttachedUserPolicies](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [ListAttachedUserPolicies](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **ListEntitiesForPolicy** dengan AWS SDK atau alat baris perintah

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 [ListEntitiesForPolicy](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [ListEntitiesForPolicy](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **ListGroupPolicies** dengan AWS SDK atau alat baris perintah

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 [ListGroupPolicies](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [ListGroupPolicies](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **ListGroups** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan `ListGroups`.

.NET

AWS SDK for .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 [ListGroups](#) di Referensi AWS SDK for .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": "AIDGPMS9R04H3FEXAMPLE",
      "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 [ListGroupsd](#) di Referensi AWS CLI Perintah.

Go

SDK for Go V2

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
// 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(maxGroups int32) ([]types.Group, error) {
    var groups []types.Group
    result, err := wrapper.IamClient.ListGroups(context.TODO(),
        &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 [ListGroups](#) di Referensi AWS SDK for 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 [ListGroups](#) di Referensi AWS SDK for JavaScript API.

PHP

SDK for 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 [ListGroups](#) di Referensi AWS SDK for PHP API.

PowerShell

Alat untuk PowerShell

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  : 6WCH4TRY3KIHIEEXAMPLE1
GroupName : Administrators
Path     : /

Arn      : arn:aws:iam::123456789012:group/Developers
CreateDate : 12/10/2014 3:38:55 PM
GroupId  : ZU2E0WMK6WBZOEXAMPLE2
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 [ListGroups](#) di Referensi AWS Tools for PowerShell Cmdlet.

Python**SDK for 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 [ListGroup](#) di AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK for 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
```

```
end
```

- Untuk detail API, lihat [ListGroups](#) di Referensi AWS SDK for 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 [ListGroups](#) referensi AWS SDK for Rust API.

Swift

SDK for Swift

Note

Ini adalah dokumentasi prarilis untuk SDK dalam rilis pratinjau. Dokumentasi ini dapat berubah.

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
public func listGroups() async throws -> [String] {
    var groupList: [String] = []
    var marker: String? = nil
    var isTruncated: Bool

    repeat {
        let input = ListGroupsInput(marker: marker)
        let output = try await client.listGroups(input: input)

        guard let groups = output.groups else {
            return groupList
        }

        for group in groups {
            if let name = group.groupName {
                groupList.append(name)
            }
        }
        marker = output.marker
        isTruncated = output.isTruncated
    } while isTruncated == true
    return groupList
}
```


- Untuk detail API, lihat referensi [ListGroups AWSSDK](#) untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **ListGroupsForUser** dengan AWS SDK atau alat baris perintah

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 grup yang menjadi nama Bob pengguna IAM.

```
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 [ListGroupForUser](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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   : 6WCH4TRY3KIHIEEXAMPLE1
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   : ZU2E0WMK6WBZOEXAMPLE3
GroupName : Developers
Path      : /
```

- Untuk detail API, lihat [ListGroupForUser](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **ListInstanceProfiles** dengan AWS SDK atau alat baris perintah

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": "AR0AJ520TH4H7LEXAMPLE",
          "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": "AIPAJVJVNRIQFEXAMPLE",
  "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": "AROAINUBC507XEXAMPLE",
      "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 [ListInstanceProfiles](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [ListInstanceProfiles](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **ListInstanceProfilesForRole** dengan AWS SDK atau alat baris perintah

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": [
    {
      "InstanceId": "AIDGPMS9R04H3FEXAMPLE",
      "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 [ListInstanceProfilesForRole](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [ListInstanceProfilesForRole](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **ListMfaDevices** dengan AWS SDK atau alat baris perintah

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 [ListMfaDevices](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [ListMfaDevices](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **ListOpenIdConnectProviders** dengan AWS SDK atau alat baris perintah

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 [ListOpenIdConnectProviders](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [ListOpenIdConnectProviders](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **ListPolicies** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan `ListPolicies`.

Contoh-contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan di dalam konteks. Anda dapat melihat tindakan ini dalam konteks pada contoh kode berikut:

- [Kelola kebijakan](#)

.NET

AWS SDK for .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 [ListPolicies](#) di Referensi AWS SDK for .NET API.

C++

SDK for 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 [ListPolicies](#) di Referensi AWS SDK for 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": "EXAMPLErZXIi0iBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQi0iA4fQ=="
  }

```

Untuk informasi selengkapnya, lihat [Kebijakan dan izin di IAM](#) dalam AWS Panduan Pengguna IAM.

- Untuk detail API, lihat [ListPolicies](#) di Referensi AWS CLI Perintah.

Go

SDK for Go V2

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```

// 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(maxPolicies int32) ([]types.Policy,
error) {
    var policies []types.Policy
    result, err := wrapper.IamClient.ListPolicies(context.TODO(),
&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 [ListPolicies](#) di Referensi AWS SDK for 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#paginator | 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 [ListPolicies](#) di Referensi AWS SDK for JavaScript API.

PHP

SDK for 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 [ListPolicies](#) di Referensi AWS SDK for PHP API.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengembalikan kumpulan dari tiga kebijakan terkelola pertama yang tersedia di AWS akun saat ini. Karena tidak **-scope** ditentukan, defaultnya **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     : 5ULJS02FYVPYGEXAMPLE3
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     : SQVCBLC4VA0UCEXAMPLE4
PolicyName   : MyLocalPolicy
UpdateDate   : 2/12/2015 9:39:53 AM

Arn           : arn:aws:iam::123456789012:policy/policyforec2instanceroles
AttachmentCount : 1
CreateDate    : 2/17/2015 2:51:38 PM
DefaultVersionId : v11
Description    :
IsAttachable  : True
Path         : /

```

```
PolicyId      : X5JPBLJH2Z2S0EXAMPLE5
PolicyName    : policyforec2instanceroles
UpdateDate   : 2/18/2015 8:52:31 AM
```

- Untuk detail API, lihat [ListPolicies](#) di Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK for 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 [ListPolicies](#) di AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK for 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 [ListPolicies](#) di Referensi AWS SDK for 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 [ListPolicies](#) referensi AWS SDK for Rust API.

Swift

SDK for Swift

Note

Ini adalah dokumentasi prarilis untuk SDK dalam rilis pratinjau. Dokumentasi ini dapat berubah.

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
public func listPolicies() async throws -> [MyPolicyRecord] {
    var policyList: [MyPolicyRecord] = []
    var marker: String? = nil
    var isTruncated: Bool

    repeat {
```

```
let input = ListPoliciesInput(marker: marker)
let output = try await client.listPolicies(input: input)

guard let policies = output.policies else {
    return policyList
}

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))
}
marker = output.marker
isTruncated = output.isTruncated
} while isTruncated == true
return policyList
}
```

- Untuk detail API, lihat referensi [ListPolicies AWSSDK](#) untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **ListPolicyVersions** dengan AWS SDK atau alat baris perintah

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 [ListPolicyVersions](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [ListPolicyVersions](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **ListRolePolicies** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan `ListRolePolicies`.

.NET

AWS SDK for .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 [ListRolePolicies](#) di Referensi AWS SDK for .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 [ListRolePolicies](#) di Referensi AWS CLI Perintah.

Go

SDK for Go V2

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
// 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(roleName string) ([]string, error) {
    var policies []string
    result, err := wrapper.IamClient.ListRolePolicies(context.TODO(),
        &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 [ListRolePolicies](#) di Referensi AWS SDK for 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 [ListRolePolicies](#) di Referensi AWS SDK for JavaScript API.

PHP

SDK for 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 [ListRolePolicies](#) di Referensi AWS SDK for PHP API.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengembalikan daftar nama kebijakan inline yang disematkan dalam peran IAM. **lambda_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 [ListRolePolicies](#) di Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK for 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 [ListRolePolicies](#) di AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK for 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 [ListRolePolicies](#) di Referensi AWS SDK for 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 [ListRolePolicies](#) referensi AWS SDK for Rust API.

Swift

SDK for Swift

Note

Ini adalah dokumentasi prarilis untuk SDK dalam rilis pratinjau. Dokumentasi ini dapat berubah.

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
public func listRolePolicies(role: String) async throws -> [String] {
    var policyList: [String] = []
    var marker: String? = nil
    var isTruncated: Bool

    repeat {
        let input = ListRolePoliciesInput(
            marker: marker,
            roleName: role
        )
        let output = try await client.listRolePolicies(input: input)

        guard let policies = output.policyNames else {
            return policyList
        }

        for policy in policies {
            policyList.append(policy)
        }
        marker = output.marker
        isTruncated = output.isTruncated
    } while isTruncated == true
    return policyList
}
```

- Untuk detail API, lihat referensi [ListRolePolicies AWSSDK](#) untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **ListRoleTags** dengan AWS SDK atau alat baris perintah

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 [ListRoleTags](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengambil tag yang terkait dengan peran..

```
Get-IAMRoleTagList -RoleName MyRoleName
```

- Untuk detail API, lihat [ListRoleTags](#) di Referensi AWS Tools for PowerShell Cmdlet.


Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **ListRoles** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan `ListRoles`.

.NET

AWS SDK for .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 [ListRoles](#) di Referensi AWS SDK for .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": "AROAJ520TH4H7LEXAMPLE",
      "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 [ListRoles](#) di Referensi AWS CLI Perintah.

Go

SDK for Go V2

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
// 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(maxRoles int32) ([]types.Role, error) {
    var roles []types.Role
    result, err := wrapper.IamClient.ListRoles(context.TODO(),
        &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 [ListRoles](#) di Referensi AWS SDK for 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 [ListRoles](#) di Referensi AWS SDK for JavaScript API.

PHP

SDK for 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 [ListRoles](#) di Referensi AWS SDK for PHP API.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengambil daftar semua peran IAM di Akun AWS

```
Get-IAMRoleList
```

Contoh 2: Contoh cuplikan kode ini mengambil daftar peran IAM di AWS akun dan menampilkannya tiga sekaligus, dan menunggu Anda menekan Enter di antara setiap grup. Ini melewati **Marker** nilai dari panggilan sebelumnya untuk menentukan di mana grup berikutnya harus dimulai.

```
$nextMarker = $null
Do
{
    $results = Get-IAMRoleList -MaxItem 3 -Marker $nextMarker
```

```
$nextMarker = $AWSHistory.LastServiceResponse.Marker
$results
Read-Host
} while ($nextMarker -ne $null)
```

- Untuk detail API, lihat [ListRoles](#) di Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK for 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 [ListRoles](#) di AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK for 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 [ListRoles](#) di Referensi AWS SDK for 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 [ListRoles](#) referensi AWS SDK for Rust API.

Swift

SDK for Swift

Note

Ini adalah dokumentasi prarilis untuk SDK dalam rilis pratinjau. Dokumentasi ini dapat berubah.

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
public func listRoles() async throws -> [String] {
    var roleList: [String] = []
    var marker: String? = nil
    var isTruncated: Bool

    repeat {
        let input = ListRolesInput(marker: marker)
        let output = try await client.listRoles(input: input)

        guard let roles = output.roles else {
            return roleList
        }

        for role in roles {
            if let name = role.roleName {
                roleList.append(name)
            }
        }
        marker = output.marker
        isTruncated = output.isTruncated
    } while isTruncated == true
    return roleList
}
```

- Untuk detail API, lihat referensi [ListRoles AWSSDK](#) untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **ListSAMLProviders** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan `ListSAMLProviders`.

.NET

AWS SDK for .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 for .NET Referensi API.

CLI

AWS CLI

Untuk 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 for Go V2

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
// 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() ([]types.SAMLProviderListEntry,
error) {
    var providers []types.SAMLProviderListEntry
    result, err := wrapper.IamClient.ListSAMLProviders(context.TODO(),
&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 for 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 for JavaScript Referensi API.

PHP

SDK for 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 for PHP Referensi API.

PowerShell

Alat untuk PowerShell

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 AWS Tools for PowerShell Referensi Cmdlet.

Python

SDK for 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 for 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 for 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.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **ListServerCertificates** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan `ListServerCertificates`.

C++

SDK for 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 [ListServerCertificates](#) di Referensi AWS SDK for 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 [ListServerCertificates](#) di 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 for JavaScript](#).
- Untuk detail API, lihat [ListServerCertificates](#) di Referensi AWS SDK for 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 for JavaScript](#).
- Untuk detail API, lihat [ListServerCertificates](#) di Referensi AWS SDK for JavaScript API.

PowerShell

Alat untuk PowerShell

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 [ListServerCertificates](#) di Referensi AWS Tools for PowerShell Cmdlet.

Ruby

SDK for 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}")
end
```

```
    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
end
```

- Untuk detail API, lihat [ListServerCertificates](#) di Referensi AWS SDK for Ruby API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **ListSigningCertificates** dengan AWS SDK atau alat baris perintah

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": "TA7SMP42TDN5Z260BPJE7EXAMPLE",
  "UploadDate": "2013-06-06T21:40:08Z"
}
```

Untuk informasi selengkapnya, lihat [Mengelola sertifikat penandatanganan](#) di Panduan Pengguna Amazon EC2.

- Untuk detail API, lihat [ListSigningCertificates](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengambil rincian tentang sertifikat penandatanganan yang terkait dengan nama **Bob** pengguna.

```
Get-IAMSigningCertificate -UserName Bob
```

Output:

```
CertificateBody : -----BEGIN CERTIFICATE-----
MIICiTCCAfICCQD6m7oRw0uX0jANBgqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
VVMxCzAJBgNVBAGTAldBMRAwDgYDVQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6
b24xFDASBgNVBAsTC01BTSBDb25zb2x1MRIwEAYDVQQDEw1UZXR0Q21sYWMxHzAd
BgkqhkiG9w0BCQEWEG5vb251QGftYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAGTAldBMRAwDgYD
VQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC01BTSBDb25z
b2x1MRIwEAYDVQQDEw1UZXR0Q21sYWMxHzAdBgkqhkiG9w0BCQEWEG5vb251QGft
```

```

YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn
+a4GmWIWJ
21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEI03IyNoH/
f0wYK8m9T
rDHudUZg3qX4waLG5M43q7Wgc/
MbQITx0USQv7c7ugFFDzQGBzZswY6786m86gpE

Ibb30hjZnzcVQAaRHhdLQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
nUhVVxYUntneD9+h8Mg9q6q
+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb

FFBjvSfpJI1J00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp3780D8uTs7fLvJx79LjSTb
NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
-----END CERTIFICATE-----
CertificateId   : Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU
Status         : Active
UploadDate    : 4/20/2015 1:26:01 PM
UserName      : Bob

```

- Untuk detail API, lihat [ListSigningCertificates](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **ListUserPolicies** dengan AWS SDK atau alat baris perintah

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 [ListUserPolicies](#) di Referensi AWS CLI Perintah.

Go

SDK for Go V2

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
// 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(userName string) ([]string, error) {
  var policies []string
  result, err := wrapper.IamClient.ListUserPolicies(context.TODO(),
    &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 [ListUserPolicies](#) di Referensi AWS SDK for Go API.

PowerShell

Alat untuk PowerShell

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 [ListUserPolicies](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **ListUserTags** dengan AWS SDK atau alat baris perintah

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 [ListUserTags](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengambil tag yang terkait dengan pengguna.

```
Get-IAMUserTagList -UserName joe
```

- Untuk detail API, lihat [ListUserTags](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **ListUsers** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan `ListUsers`.

Contoh-contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan di dalam konteks. Anda dapat melihat tindakan ini dalam konteks pada contoh kode berikut:

- [Buat pengguna read-only dan read-write](#)

.NET

AWS SDK for .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 [ListUsers](#) di Referensi AWS SDK for .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 getopt 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 [ListUsers](#) di Referensi AWS CLI Perintah.

C++

SDK for 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 [ListUsers](#) di Referensi AWS SDK for 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 [ListUsers](#) di Referensi AWS CLI Perintah.

Go

SDK for Go V2

 Note

Ada lebih banyak tentang GitHub. Temukan contoh selengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
// 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(maxUsers int32) ([]types.User, error) {
    var users []types.User
    result, err := wrapper.IamClient.ListUsers(context.TODO(), &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 [ListUsers](#) di Referensi AWS SDK for Go API.

Java

SDK for 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 [ListUsers](#) di 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);
  response.Users?.forEach(({ UserName, CreateDate }) => {
    console.log(`${UserName} created on: ${CreateDate}`);
  });
  return response;
};
```

- Untuk informasi selengkapnya, silakan lihat [Panduan Developer AWS SDK for JavaScript](#).
- Untuk detail API, lihat [ListUsers](#) di Referensi AWS SDK for 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 for JavaScript](#).
- Untuk detail API, lihat [ListUsers](#) di Referensi AWS SDK for JavaScript API.

Kotlin

SDK for 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 { 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 [ListUsers](#) di AWS SDK untuk referensi API Kotlin.

PHP

SDK for 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 [ListUsers](#) di Referensi AWS SDK for PHP API.

PowerShell

Alat untuk PowerShell

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 [ListUsers](#) di Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK for 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 [ListUsers](#) di AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK for 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
end
users
```

```
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error listing users: #{e.message}")
  []
end
```

- Untuk detail API, lihat [ListUsers](#) di Referensi AWS SDK for 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 [ListUsers](#) referensi AWS SDK for Rust API.

Swift

SDK for Swift

Note

Ini adalah dokumentasi prarilis untuk SDK dalam rilis pratinjau. Dokumentasi ini dapat berubah.

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
public func listUsers() async throws -> [MyUserRecord] {
    var userList: [MyUserRecord] = []
    var marker: String? = nil
    var isTruncated: Bool

    repeat {
        let input = ListUsersInput(marker: marker)
        let output = try await client.listUsers(input: input)

        guard let users = output.users else {
            return userList
        }

        for user in users {
            if let id = user.userId, let name = user.userName {
                userList.append(MyUserRecord(id: id, name: name))
            }
        }
        marker = output.marker
        isTruncated = output.isTruncated
    } while isTruncated == true
    return userList
}
```

- Untuk detail API, lihat referensi [ListUsers AWSSDK](#) untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **ListVirtualMfaDevices** dengan AWS SDK atau alat baris perintah

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 [ListVirtualMfaDevices](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [ListVirtualMfaDevices](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **PutGroupPolicy** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan `PutGroupPolicy`.

Contoh-contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan di dalam konteks. Anda dapat melihat tindakan ini dalam konteks pada contoh kode berikut:

- [Buat grup dan tambahkan pengguna](#)

.NET

AWS SDK for .NET

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
/// <summary>
/// Add or update an inline policy document that is embedded in an IAM group.
/// </summary>
/// <param name="groupName">The name of the IAM group.</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> PutGroupPolicyAsync(string groupName, string
policyName, string policyDocument)
{
    var request = new PutGroupPolicyRequest
    {
        GroupName = groupName,
        PolicyName = policyName,
        PolicyDocument = policyDocument
    };

    var response = await _IAMService.PutGroupPolicyAsync(request);
    return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
}
```

- Untuk detail API, lihat [PutGroupPolicy](#) di Referensi AWS SDK for .NET API.

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 `AdminPolicyfile.json`. (Nama file dan ekstensi tidak memiliki signifikansi.)

Untuk informasi selengkapnya, lihat [Mengelola kebijakan IAM](#) di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat [PutGroupPolicy](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [PutGroupPolicy](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **PutRolePermissionsBoundary** dengan AWS SDK atau alat baris perintah

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 [PutRolePermissionsBoundary](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [PutRolePermissionsBoundary](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **PutRolePolicy** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan `PutRolePolicy`.

.NET

AWS SDK for .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 [PutRolePolicy](#) di Referensi AWS SDK for .NET API.

C++

SDK for 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 [PutRolePolicy](#) di Referensi AWS SDK for 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 `AdminPolicyfile.json`. (Nama file dan ekstensi tidak memiliki signifikansi.)

Untuk melampirkan kebijakan kepercayaan ke peran, gunakan `update-assume-role-policy` perintah.

Untuk informasi selengkapnya, lihat [Memodifikasi peran](#) dalam Panduan Pengguna AWS IAM.

- Untuk detail API, lihat [PutRolePolicy](#) di 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:::some-test-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 [PutRolePolicy](#) di Referensi AWS SDK for JavaScript API.

PowerShell

Alat untuk PowerShell

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 [PutRolePolicy](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **PutUserPermissionsBoundary** dengan AWS SDK atau alat baris perintah

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 policy AWS terkelola bernama `PowerUserAccess` sebagai batas izin untuk pengguna IAM yang ditentukan.

```
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 [PutUserPermissionsBoundary](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [PutUserPermissionsBoundary](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **PutUserPolicy** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan `PutUserPolicy`.

Contoh-contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan di dalam konteks. Anda dapat melihat tindakan ini dalam konteks pada contoh kode berikut:

- [Buat pengguna dan ambil peran](#)

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 `AdminPolicyfile.json`. (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 [PutUserPolicy](#) di Referensi AWS CLI Perintah.

Go

SDK for Go V2

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
// 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(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(context.TODO(),
        &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 [PutUserPolicy](#) di Referensi AWS SDK for Go API.

PowerShell

Alat untuk PowerShell

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 [PutUserPolicy](#) di Referensi AWS Tools for PowerShell Cmdlet.

Ruby

SDK for 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}.")
end
```

```
    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 [PutUserPolicy](#) di Referensi AWS SDK for Ruby API.

Swift

SDK for Swift

Note

Ini adalah dokumentasi prarilis untuk SDK dalam rilis pratinjau. Dokumentasi ini dapat berubah.

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
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 {
        throw error
    }
}
```

- Untuk detail API, lihat referensi [PutUserPolicy AWSSDK](#) untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **RemoveClientIdFromOpenIdConnectProvider** dengan AWS SDK atau alat baris perintah

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 [RemoveClientIdFromOpenIdConnectProvider](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: 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**

```
Remove-IAMClientIDFromOpenIDConnectProvider -ClientID My-TestApp-3
-OpenIDConnectProviderArn arn:aws:iam::123456789012:oidc-provider/
example.oidcprovider.com
```

- Untuk detail API, lihat [RemoveClientIDFromOpenIDConnectProvider](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **RemoveRoleFromInstanceProfile** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan `RemoveRoleFromInstanceProfile`.

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 [RemoveRoleFromInstanceProfile](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [RemoveRoleFromInstanceProfile](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **RemoveUserFromGroup** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan `RemoveUserFromGroup`.

Contoh-contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan di dalam konteks. Anda dapat melihat tindakan ini dalam konteks pada contoh kode berikut:

- [Buat grup dan tambahkan pengguna](#)

.NET

AWS SDK for .NET

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
/// <summary>
```

```
/// Remove a user from an IAM group.
/// </summary>
/// <param name="userName">The username of the user to remove.</param>
/// <param name="groupName">The name of the IAM group to remove the user
from.</param>
/// <returns>A Boolean value indicating the success of the action.</returns>
public async Task<bool> RemoveUserFromGroupAsync(string userName, string
groupName)
{
    // Remove the user from the group.
    var removeUserRequest = new RemoveUserFromGroupRequest()
    {
        UserName = userName,
        GroupName = groupName,
    };

    var response = await
_IAMService.RemoveUserFromGroupAsync(removeUserRequest);
    return response.HttpStatusCode == HttpStatusCode.OK;
}
```

- Untuk detail API, lihat [RemoveUserFromGroup](#) di Referensi AWS SDK for .NET API.

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 [RemoveUserFromGroup](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [RemoveUserFromGroup](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **ResyncMfaDevice** dengan AWS SDK atau alat baris perintah

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 [ResyncMfaDevice](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [ResyncMfaDevice](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **SetDefaultPolicyVersion** dengan AWS SDK atau alat baris perintah

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 [SetDefaultPolicyVersion](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini menetapkan **v2** versi kebijakan yang ARN `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 [SetDefaultPolicyVersion](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **TagRole** dengan AWS SDK atau alat baris perintah

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 [TagRole](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini menambahkan tag ke Peran dalam Layanan Manajemen Identitas

```
Add-IAMRoleTag -RoleName AdminRoleaccess -Tag @{ Key = 'abac'; Value = 'testing'}
```

- Untuk detail API, lihat [TagRole](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **TagUser** dengan AWS SDK atau alat baris perintah

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 [TagUser](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [TagUser](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **UntagRole** dengan AWS SDK atau alat baris perintah

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 [UntagRole](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [UntagRole](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **UntagUser** dengan AWS SDK atau alat baris perintah

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 [UntagUser](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [UntagUser](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **UpdateAccessKey** dengan AWS SDK atau alat baris perintah


Contoh kode berikut menunjukkan cara menggunakan `UpdateAccessKey`.

Contoh-contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan di dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- [Kelola kunci akses](#)

C++

SDK for 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 [UpdateAccessKey](#) di Referensi AWS SDK for 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 [UpdateAccessKey](#) di Referensi AWS CLI Perintah.

Java

SDK for 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 [UpdateAccessKey](#) di 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 for JavaScript](#).
- Untuk detail API, lihat [UpdateAccessKey](#) di Referensi AWS SDK for 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 for JavaScript](#).
- Untuk detail API, lihat [UpdateAccessKey](#) di Referensi AWS SDK for JavaScript API.

PowerShell

Alat untuk PowerShell

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 [UpdateAccessKey](#) di Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK for 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 [UpdateAccessKey](#) di AWS SDK for Python (Boto3) Referensi API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **UpdateAccountPasswordPolicy** dengan AWS SDK atau alat baris perintah

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 [UpdateAccountPasswordPolicy](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [UpdateAccountPasswordPolicy](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **UpdateAssumeRolePolicy** dengan AWS SDK atau alat baris perintah

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 file `test-role-trust-policy.json`. (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 [UpdateAssumeRolePolicy](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [UpdateAssumeRolePolicy](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **UpdateGroup** dengan AWS SDK atau alat baris perintah

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 [UpdateGroup](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [UpdateGroup](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **UpdateLoginProfile** dengan AWS SDK atau alat baris perintah

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 [UpdateLoginProfile](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [UpdateLoginProfile](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan `UpdateOpenIdConnectProviderThumbprint` dengan AWS SDK atau alat baris perintah

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
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat [Membuat penyedia identitas OpenID Connect \(OIDC\)](#) di Panduan Pengguna IAM.AWS

- Untuk detail API, lihat [UpdateOpenIdConnectProviderThumbprint](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [UpdateOpenIdConnectProviderThumbprint](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **UpdateRole** dengan AWS SDK atau alat baris perintah

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 [UpdateRole](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [UpdateRole](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **UpdateRoleDescription** dengan AWS SDK atau alat baris perintah

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 [UpdateRoleDescription](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini memperbarui deskripsi peran IAM di akun Anda.

```
Update-IAMRoleDescription -RoleName MyRoleName -Description "My testing role"
```

- Untuk detail API, lihat [UpdateRoleDescription](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **UpdateSamlProvider** dengan AWS SDK atau alat baris perintah

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 [UpdateSamlProvider](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [UpdateSamlProvider](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **UpdateServerCertificate** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan `UpdateServerCertificate`.

C++

SDK for 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 [UpdateServerCertificate](#) di Referensi AWS SDK for 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 [UpdateServerCertificate](#) di 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 for JavaScript](#).
- Untuk detail API, lihat [UpdateServerCertificate](#) di Referensi AWS SDK for 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 for JavaScript](#).
- Untuk detail API, lihat [UpdateServerCertificate](#) di Referensi AWS SDK for JavaScript API.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengganti nama sertifikat yang dinamai **MyServerCertificate** menjadi **MyRenamedServerCertificate**

```
Update-IAMServerCertificate -ServerCertificateName MyServerCertificate -
NewServerCertificateName MyRenamedServerCertificate
```

Contoh 2: Contoh ini memindahkan sertifikat bernama **MyServerCertificate** ke path **/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 [UpdateServerCertificate](#) di Referensi AWS Tools for PowerShell Cmdlet.

Ruby

SDK for 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
  end
end
```

```
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
end
```

- Untuk detail API, lihat [UpdateServerCertificate](#) di Referensi AWS SDK for Ruby API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan `UpdateSigningCertificate` dengan AWS SDK atau alat baris perintah

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 TA7SMP42TDN5Z2260BPJE7EXAMPLE \  
  --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 [UpdateSigningCertificate](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [UpdateSigningCertificate](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **UpdateUser** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan `UpdateUser`.

Contoh-contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan di dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- [Buat pengguna read-only dan read-write](#)

C++

SDK for 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 [UpdateUser](#) di Referensi AWS SDK for 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 [UpdateUser](#) di Referensi AWS CLI Perintah.

Java

SDK for 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 [UpdateUser](#) di 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: currentUser,
  NewUserName: newUserName,
});

return client.send(command);
};
```

- Untuk informasi selengkapnya, silakan lihat [Panduan Developer AWS SDK for JavaScript](#).
- Untuk detail API, lihat [UpdateUser](#) di Referensi AWS SDK for 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 for JavaScript](#).
- Untuk detail API, lihat [UpdateUser](#) di Referensi AWS SDK for JavaScript API.

Kotlin

SDK for 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 { region = "AWS_GLOBAL" }.use { iamClient ->  
        iamClient.updateUser(request)  
        println("Successfully updated user to $newName")  
    }  
}
```

- Untuk detail API, lihat [UpdateUser](#) di AWS SDK untuk referensi API Kotlin.

PowerShell

Alat untuk PowerShell

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 [UpdateUser](#) di Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK for 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 [UpdateUser](#) di AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK for 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 [UpdateUser](#) di Referensi AWS SDK for Ruby API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **UploadServerCertificate** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan `UploadServerCertificate`.

CLI

AWS CLI

Untuk mengunggah sertifikat server ke AWS akun Anda

`upload-server-certificate` Perintah berikut mengunggah sertifikat server ke AWS akun Anda. 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 di bawah nama `myServerCertificate`. 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 [UploadServerCertificate](#) di 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 "fs";
import { dirnameFromMetaUrl } from "@aws-doc-sdk-examples/lib/utils/util-fs.js";
import * as path from "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 [UploadServerCertificate](#) di Referensi AWS SDK for JavaScript API.

PowerShell

Alat untuk PowerShell

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 [UploadServerCertificate](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **UploadSigningCertificate** dengan AWS SDK atau alat baris perintah

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": "TA7SMP42TDN5Z260BPJE7EXAMPLE",  
    "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 [UploadSigningCertificate](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengunggah sertifikat penandatanganan X.509 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-----

MIICiTCCAFICCCQD6m7oRw0uX0jANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC

VVMxCzAJBgNVBAGTAldBMRAwDgYDVQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6

b24xFDASBgNVBA5TC01BTSBDb25zb2x1MRIwEAYDVQQDEw1UZXR0Q21sYWxhZAd

BgkqhkiG9w0BCQEWEG5vb25lQGFTtYXpvi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN

MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCMVVMxCzAJBgNVBAGTAldBMRAwDgYD

VQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBA5TC01BTSBDb25z

b2x1MRIwEAYDVQQDEw1UZXR0Q21sYWxhZAdBgkqhkiG9w0BCQEWEG5vb25lQGFT

YXpvi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn

+a4GmWIWJ

21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEI03IyNoH/

f0wYK8m9T

rDHudUZg3qX4waLG5M43q7Wgc/

MbQITx0USQv7c7ugFFDzQGBzZswY6786m86gpE

Ibb30hjZnzcvcQAaRHhd1QWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4

nUhVVxYUntneD9+h8Mg9q6q

+auNKyExzyLwax1Aoo7TJHidbtS4J5iNmZgXL0Fkb
```

```
FFBjvSfpJI1J00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp3780D8uTs7fLvJx79LjSTb
      NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
      -----END CERTIFICATE-----
CertificateId   : Y3EK7RMEXAMPLESV33FCEXAMPLEHMJLU
Status         : Active
UploadDate    : 4/20/2015 1:26:01 PM
UserName      : Bob
```

- Untuk detail API, lihat [UploadSigningCertificate](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Skenario untuk IAM menggunakan 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. Setiap skenario menyertakan tautan ke GitHub, di mana Anda dapat menemukan petunjuk tentang cara mengatur dan menjalankan kode.

Contoh-contoh

- [Membangun dan mengelola layanan tangguh menggunakan SDK AWS](#)
- [Buat grup IAM dan tambahkan pengguna ke grup menggunakan SDK AWS](#)
- [Buat pengguna IAM dan ambil peran dengan AWS STS menggunakan SDK AWS](#)
- [Buat pengguna IAM read-only dan read-write menggunakan SDK AWS](#)
- [Mengelola kunci akses IAM menggunakan SDK AWS](#)
- [Mengelola kebijakan IAM menggunakan SDK AWS](#)
- [Mengelola peran IAM menggunakan SDK AWS](#)
- [Mengelola akun IAM Anda menggunakan SDK AWS](#)
- [Kembalikan versi kebijakan IAM menggunakan SDK AWS](#)
- [Bekerja dengan API Pembuat Kebijakan IAM menggunakan SDK AWS](#)

Membangun dan mengelola layanan tangguh menggunakan SDK AWS

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

AWS SDK for .NET

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan 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.TargetGroup
        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($"http://{endPoint}\n");
}
else
{
    Console.WriteLine(
        "\nCouldn't get a successful response from the load balancer
endpoint. Troubleshoot by\n"
        + "manually verifying that your VPC and security group are
configured correctly and that\n"
        + "you can successfully make a GET request to the load balancer
endpoint:\n");
    Console.WriteLine($"http://{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($"Now, 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($"Even 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($"When 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 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)
{
    _amazonAutoScaling = amazonAutoScaling;
    _amazonEc2 = amazonEc2;
    _amazonSsm = amazonSsm;
    _amazonIam = amazonIam;

    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="newKeyName">The name of the new key pair.</param>
/// <returns>Async task.</returns>
public async Task CreateKeyPair(string newKeyName)
{
    try
    {
        var keyResponse = await _amazonEc2.CreateKeyPairAsync(
            new CreateKeyPairRequest() { KeyName = newKeyName });
        await File.WriteAllTextAsync($"{newKeyName}.pem",
            keyResponse.KeyPair.KeyMaterial);
        Console.WriteLine($"Created key pair {newKeyName}.");
    }
    catch (AlreadyExistsException)
    {
        Console.WriteLine("Key pair already exists.");
    }
}

/// <summary>
/// Delete the key pair and file by name.
/// </summary>
/// <param name="deleteKeyName">The key pair to delete.</param>
/// <returns>Async task.</returns>
public async Task DeleteKeyPairByName(string deleteKeyName)
{
    try
    {
        await _amazonEc2.DeleteKeyPairAsync(
            new DeleteKeyPairRequest() { KeyName = deleteKeyName });
    }
}
```

```
        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)
{
    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;
}

/// <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()
{
    var zoneResponse = await _amazonEc2.DescribeAvailabilityZonesAsync(
        new DescribeAvailabilityZonesRequest());
    return zoneResponse.AvailabilityZones.Select(z => z.ZoneName).ToList();
}

/// <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()
{
    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];
}

/// <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)
{
    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;
}

/// <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 (AmazonClientException)
    {
        Console.WriteLine($"Unable to delete template {templateName}.");
    }
}
```

```
/// <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)
{
    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];
    }

    /// <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)
    {
        await _amazonEc2.ReplaceIamInstanceProfileAssociationAsync(
            new ReplaceIamInstanceProfileAssociationRequest()
            {
                AssociationId = associationId,
                IamInstanceProfile = new IamInstanceProfileSpecification()
                {
                    Name = credsProfileName
                }
            }
        );
        // Allow time before resetting.
        Thread.Sleep(25000);
        var instanceReady = false;
        var retries = 5;
        while (retries-- > 0 && !instanceReady)
        {
            await _amazonEc2.RebootInstancesAsync(
                new RebootInstancesRequest(new List<string>() { instanceId }));
            Thread.Sleep(10000);

            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($"Sending restart command to instance {instanceId}");
    await _amazonSsm.SendCommandAsync(
        new SendCommandRequest()
        {
            InstanceIds = new List<string>() { instanceId },
            DocumentName = "AWS-RunShellScript",
            Parameters = new Dictionary<string, List<string>>()
            {
                {"commands", new List<string>() { "cd / && sudo python3
server.py 80" }}
            }
        });
    Console.WriteLine($"Restarted the web server on instance {instanceId}");
}

/// <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
                }
            );
        }
        catch { }
    }
}
```

```
        });
        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 }
        });
}
}
}

```

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 mensimulasikan 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.WriteLine($"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.WriteLine("\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.WriteLine(".");
    }
    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 menggabungkan 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 for .NET .
 - [AttachLoadBalancerTargetGroups](#)
 - [CreateAutoScalingGroup](#)

- [CreateInstanceProfile](#)
- [CreateLaunchTemplate](#)
- [CreateListener](#)
- [CreateLoadBalancer](#)
- [CreateTargetGroup](#)
- [DeleteAutoScalingGroup](#)
- [DeleteInstanceProfile](#)
- [DeleteLaunchTemplate](#)
- [DeleteLoadBalancer](#)
- [DeleteTargetGroup](#)
- [DescribeAutoScalingGroups](#)
- [DescribeAvailabilityZones](#)
- [DescribeIamInstanceProfileAssociations](#)
- [DescribeInstances](#)
- [DescribeLoadBalancers](#)
- [DescribeSubnets](#)
- [DescribeTargetGroups](#)
- [DescribeTargetHealth](#)
- [DescribeVpcs](#)
- [RebootInstances](#)
- [ReplacelamInstanceProfileAssociation](#)
- [TerminateInstanceInAutoScalingGroup](#)
- [UpdateAutoScalingGroup](#)

Java

SDK for Java 2.x

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya 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 wasSuccessful =
        loadBalancer.verifyLoadBalancerEndpoint(elbDnsName);
        if (!wasSuccessful) {
```



```
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 (wasSuccessful) {
        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.getInstanceInformationList();
    for (InstanceInformation info : instanceInformationList) {
        if (info.getInstanceId().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.getInstanceProfile().getInstanceProfileName();
            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)
    .defaultActions(action)
    .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 mensimulasikan 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 menggabungkan 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](#)
- [DescribelamInstanceProfileAssociations](#)
- [DescribeInstances](#)
- [DescribeLoadBalancers](#)
- [DescribeSubnets](#)
- [DescribeTargetGroups](#)
- [DescribeTargetHealth](#)
- [DescribeVpcs](#)
- [RebootInstances](#)
- [ReplacelamInstanceProfileAssociation](#)
- [TerminateInstanceInAutoScalingGroup](#)
- [UpdateAutoScalingGroup](#)

JavaScript

SDK untuk JavaScript (v3)

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya 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 "url";

if (process.argv[1] === fileURLToPath(import.meta.url)) {
```



```
    parseScenarioArgs(scenarios);
  }
```

Menyusun langkah-langkah untuk men-deploy semua sumber daya.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
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 { 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 () => {
  // snippet-start:[javascript.v3.wkflw.resilient.CreateLaunchTemplate]
  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,
      },
    }),
  );
  // snippet-end:[javascript.v3.wkflw.resilient.CreateLaunchTemplate]
});
```

```

    }),
    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) => {
        // snippet-start:[javascript.v3.wkflw.resilient.DescribeVpcs]
        const client = new EC2Client({});
        const { Vpcs } = await client.send(
            new DescribeVpcsCommand({
                Filters: [{ Name: "is-default", Values: ["true"] }],
            }),
        );
        // snippet-end:[javascript.v3.wkflw.resilient.DescribeVpcs]
        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) => {
        // snippet-start:[javascript.v3.wkflw.resilient.DescribeSubnets]
        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"] },
                ],
            }),
        );
        // snippet-end:[javascript.v3.wkflw.resilient.DescribeSubnets]
        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) => {
  // snippet-start:[javascript.v3.wkflw.resilient.CreateTargetGroup]
  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,
    })),
  );
  // snippet-end:[javascript.v3.wkflw.resilient.CreateTargetGroup]
  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) => {
  // snippet-start:[javascript.v3.wkflw.resilient.CreateLoadBalancer]
  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] },
  );
  // snippet-end:[javascript.v3.wkflw.resilient.CreateLoadBalancer]
}),
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) => {
  // snippet-start:[javascript.v3.wkflw.resilient.CreateListener]
  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 },
      ],
    }),
  );
  // snippet-end:[javascript.v3.wkflw.resilient.CreateListener]
  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) => {
    // snippet-start:[javascript.v3.wkflw.resilient.AttachTargetGroup]
    const client = new AutoScalingClient({});
    await client.send(
      new AttachLoadBalancerTargetGroupsCommand({
        AutoScalingGroupName: NAMES.autoScalingGroupName,
        TargetGroupARNs: [state.targetGroupArn],
      }),
    );
    // snippet-end:[javascript.v3.wkflw.resilient.AttachTargetGroup]
  }),
  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),
      );
    } else {
      return MESSAGES.noIpRules;
    }
  },
),
new ScenarioInput(
  "shouldAddInboundRule",
  /**
   * @param {{ myIpRules: any[] }} state
   */
```

```

    (state) => {
      if (state.myIpRules.length > 0) {
        return false;
      } else {
        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);
    } else {
      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,
    );
  }
}),
];
```

Menyusun langkah-langkah untuk menjalankan demo.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
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) => {
  // snippet-start:[javascript.v3.wkflw.resilient.DescribeTargetGroups]
  const client = new ElasticLoadBalancingV2Client({});
  const { TargetGroups } = await client.send(
    new DescribeTargetGroupsCommand({
      Names: [NAMES.loadBalancerTargetGroupName],
    }),
  );
  // snippet-end:[javascript.v3.wkflw.resilient.DescribeTargetGroups]

  // snippet-start:[javascript.v3.wkflw.resilient.DescribeTargetHealth]
  const { TargetHealthDescriptions } = await client.send(
    new DescribeTargetHealthCommand({
      TargetGroupArn: TargetGroups[0].TargetGroupArn,
    }),
  );
  // snippet-end:[javascript.v3.wkflw.resilient.DescribeTargetHealth]
  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: {
      inputEquals: true,
      input: new ScenarioInput(
        "loadBalancerCheck",
        MESSAGES.demoLoadBalancerCheck,
        {
          type: "confirm",
        },
      ),
      output: getRecommendationResult,
    },
  },
);

const healthCheckLoop = new ScenarioAction(
  "healthCheckLoop",
  getHealthCheck.action,
  {
    whileConfig: {
      inputEquals: true,
      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];
// snippet-start:
[javascript.v3.wkflw.resilient.DescribeIamInstanceProfileAssociations]
const ec2Client = new EC2Client({});
const { IamInstanceProfileAssociations } = await ec2Client.send(
  new DescribeIamInstanceProfileAssociationsCommand({
    Filters: [
      { Name: "instance-id", Values: [state.targetInstance.InstanceId] },
    ],
  }),
);
// snippet-end:
[javascript.v3.wkflw.resilient.DescribeIamInstanceProfileAssociations]
state.instanceProfileAssociationId =
  IamInstanceProfileAssociations[0].AssociationId;
// snippet-start:
[javascript.v3.wkflw.resilient.ReplaceIamInstanceProfileAssociation]
await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
  ec2Client.send(
    new ReplaceIamInstanceProfileAssociationCommand({
      AssociationId: state.instanceProfileAssociationId,
      IamInstanceProfile: { Name: NAMES.ssmOnlyInstanceProfileName },
    }),
  ),
);
// snippet-end:
[javascript.v3.wkflw.resilient.ReplaceIamInstanceProfileAssociation]

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",
    }),
  );
  // snippet-start:[javascript.v3.wkflw.resilient.CreateInstanceProfile]
  const { InstanceProfile } = await iamClient.send(
```

```
    new CreateInstanceProfileCommand({
      InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
    }),
  );
  await waitUntilInstanceProfileExists(
    { client: iamClient },
    { InstanceProfileName: NAMES.ssmOnlyInstanceProfileName },
  );
  // snippet-end:[javascript.v3.wkflw.resilient.CreateInstanceProfile]
  await iamClient.send(
    new AddRoleToInstanceProfileCommand({
      InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
      RoleName: NAMES.ssmOnlyRoleName,
    }),
  );

  return InstanceProfile;
}
```

Menyusun langkah-langkah untuk menghancurkan semua sumber daya.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
import { unlinkSync } from "node:fs";

import { DynamoDBClient, DeleteTableCommand } from "@aws-sdk/client-dynamodb";
import {
  EC2Client,
  DeleteKeyPairCommand,
  DeleteLaunchTemplateCommand,
} 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 { 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 = [
  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,
    );
  } else {
    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(
      "${KEY_PAIR_NAME}",
      NAMES.keyPairName,
    );
  } else {
    return MESSAGES.deletedKeyPair.replace(
      "${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);
  } else {
    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,
    );
  } else {
```

```
    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("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
  } else {
    return MESSAGES.removedRoleFromInstanceProfile
      .replace("${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(
        "${INSTANCE_ROLE_NAME}",
        NAMES.instanceRoleName,
      );
    } else {
      return MESSAGES.deletedInstanceRole.replace(
        "${INSTANCE_ROLE_NAME}",
        NAMES.instanceRoleName,
      );
    }
  })),
  new ScenarioAction("deleteInstanceProfile", async (state) => {
    try {
      // snippet-start:[javascript.v3.wkflw.resilient.DeleteInstanceProfile]
      const client = new IAMClient({});
      await client.send(
        new DeleteInstanceProfileCommand({
          InstanceProfileName: NAMES.instanceProfileName,
        }),
      );
      // snippet-end:[javascript.v3.wkflw.resilient.DeleteInstanceProfile]
    } catch (e) {
      state.deleteInstanceProfileError = e;
    }
  })),
  new ScenarioOutput("deleteInstanceProfileResult", (state) => {
    if (state.deleteInstanceProfileError) {
      console.error(state.deleteInstanceProfileError);
      return MESSAGES.deleteInstanceProfileError.replace(
        "${INSTANCE_PROFILE_NAME}",
        NAMES.instanceProfileName,
      );
    } else {
      return MESSAGES.deletedInstanceProfile.replace(
        "${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(
      "${AUTO_SCALING_GROUP_NAME}",
      NAMES.autoScalingGroupName,
    );
  } else {
    return MESSAGES.deletedAutoScalingGroup.replace(
      "${AUTO_SCALING_GROUP_NAME}",
      NAMES.autoScalingGroupName,
    );
  }
}),
new ScenarioAction("deleteLaunchTemplate", async (state) => {
  const client = new EC2Client({});
  try {
    // snippet-start:[javascript.v3.wkflw.resilient.DeleteLaunchTemplate]
    await client.send(
      new DeleteLaunchTemplateCommand({
        LaunchTemplateName: NAMES.launchTemplateName,
      }),
    );
    // snippet-end:[javascript.v3.wkflw.resilient.DeleteLaunchTemplate]
  } 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,
    );
  } else {
    return MESSAGES.deletedLaunchTemplate.replace(
```



```
    "${LAUNCH_TEMPLATE_NAME}",
    NAMES.launchTemplateName,
  );
}
}),
new ScenarioAction("deleteLoadBalancer", async (state) => {
  try {
    // snippet-start:[javascript.v3.wkflw.resilient.DeleteLoadBalancer]
    const client = new ElasticLoadBalancingV2Client({});
    const loadBalancer = await findLoadBalancer(NAMES.loadBalancerName);
    await client.send(
      new DeleteLoadBalancerCommand({
        LoadBalancerArn: loadBalancer.LoadBalancerArn,
      }),
    );
    await retry({ intervalInMs: 1000, maxRetries: 60 }, async () => {
      const lb = await findLoadBalancer(NAMES.loadBalancerName);
      if (lb) {
        throw new Error("Load balancer still exists.");
      }
    });
    // snippet-end:[javascript.v3.wkflw.resilient.DeleteLoadBalancer]
  } catch (e) {
    state.deleteLoadBalancerError = e;
  }
}),
new ScenarioOutput("deleteLoadBalancerResult", (state) => {
  if (state.deleteLoadBalancerError) {
    console.error(state.deleteLoadBalancerError);
    return MESSAGES.deleteLoadBalancerError.replace(
      "${LB_NAME}",
      NAMES.loadBalancerName,
    );
  } else {
    return MESSAGES.deletedLoadBalancer.replace(
      "${LB_NAME}",
      NAMES.loadBalancerName,
    );
  }
}),
new ScenarioAction("deleteLoadBalancerTargetGroup", async (state) => {
  // snippet-start:[javascript.v3.wkflw.resilient.DeleteTargetGroup]
  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;
}
// snippet-end:[javascript.v3.wkflw.resilient.DeleteTargetGroup]
}),
new ScenarioOutput("deleteLoadBalancerTargetGroupResult", (state) => {
  if (state.deleteLoadBalancerTargetGroupError) {
    console.error(state.deleteLoadBalancerTargetGroupError);
    return MESSAGES.deleteLoadBalancerTargetGroupError.replace(
      "${TARGET_GROUP_NAME}",
      NAMES.loadBalancerTargetGroupName,
    );
  } else {
    return MESSAGES.deletedLoadBalancerTargetGroup.replace(
      "${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);
      } else {
        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);
      } else {
        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");
    } else {
        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(
            "${INSTANCE_PROFILE_NAME}",
            NAMES.ssmOnlyInstanceProfileName,
        );
    } else {
        return MESSAGES.deletedSsmOnlyInstanceProfile.replace(
            "${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(
        "${POLICY_NAME}",
        NAMES.ssmOnlyPolicyName,
      );
    } else {
      return MESSAGES.deletedSsmOnlyPolicy.replace(
        "${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(
            "${ROLE_NAME}",
            NAMES.ssmOnlyRoleName,
        );
    } else {
        return MESSAGES.deletedSsmOnlyRole.replace(
            "${ROLE_NAME}",
            NAMES.ssmOnlyRoleName,
        );
    }
    })),
];

/**
 * @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;
        } else {
            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 for JavaScript .

- [AttachLoadBalancerTargetGroups](#)
- [CreateAutoScalingGroup](#)
- [CreateInstanceProfile](#)
- [CreateLaunchTemplate](#)
- [CreateListener](#)
- [CreateLoadBalancer](#)
- [CreateTargetGroup](#)
- [DeleteAutoScalingGroup](#)
- [DeleteInstanceProfile](#)
- [DeleteLaunchTemplate](#)
- [DeleteLoadBalancer](#)
- [DeleteTargetGroup](#)
- [DescribeAutoScalingGroups](#)
- [DescribeAvailabilityZones](#)
- [DescribeIamInstanceProfileAssociations](#)
- [DescribeInstances](#)
- [DescribeLoadBalancers](#)
- [DescribeSubnets](#)
- [DescribeTargetGroups](#)
- [DescribeTargetHealth](#)
- [DescribeVpcs](#)
- [RebootInstances](#)
- [ReplacelamInstanceProfileAssociation](#)
- [TerminateInstanceInAutoScalingGroup](#)
- [UpdateAutoScalingGroup](#)

Python

SDK for Python (Boto3)

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

Menjalankan skenario interaktif di prompt perintah.

```
class Runner:
    def __init__(
        self, resource_path, recommendation, autoscaler, loadbalancer,
        param_helper
    ):
        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

    def deploy(self):
        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"

        print(
            "\nFor this demo, we'll use the AWS SDK for Python (Boto3) 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"
        )
        print(
            "\t* A DynamoDB table that the web service depends on to provide
            book, movie, and song recommendations."
        )
```

```
print(
    "\t* An EC2 launch template that defines EC2 instances that each
contain a Python web server."
)
print(
    "\t* An EC2 Auto Scaling group that manages EC2 instances across
several Availability Zones."
)
print(
    "\t* An Elastic Load Balancing (ELB) load balancer that targets the
Auto Scaling group to distribute requests."
)
print("-" * 88)
q.ask("Press Enter when you're ready to start deploying resources.")

print(
    f"Creating and populating a DynamoDB table named
'{self.recommendation.table_name}'."
)
self.recommendation.create()
self.recommendation.populate(recommendations_path)
print("-" * 88)

print(
    f"Creating an EC2 launch template that runs '{startup_script}' when
an instance starts.\n"
    f"This script starts a Python web server defined in the `server.py`
script. The web server\n"
    f"listens to HTTP requests on port 80 and responds to requests to '/'
and to '/healthcheck'.\n"
    f"For demo purposes, this server is run as the root user. In
production, the best practice is to\n"
    f"run a web server, such as Apache, with least-privileged
credentials.\n"
)
print(
    f"The template also defines an IAM policy that each instance uses to
assume a role that grants\n"
    f"permissions to access the DynamoDB recommendation table and Systems
Manager parameters\n"
    f"that control the flow of the demo.\n"
)
self.autoscaler.create_template(startup_script, instance_policy)
print("-" * 88)
```

```
print(
    f"Creating an EC2 Auto Scaling group that maintains three EC2
instances, each in a different\n"
    f"Availability Zone."
)
zones = self.autoscaler.create_group(3)
print("-" * 88)
print(
    "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."
)
print("-" * 88)
q.ask("Press Enter when you're ready to continue.")

print(f"Creating variables that control the flow of the demo.\n")
self.param_helper.reset()

print(
    "\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.\n"
)
vpc = self.autoscaler.get_default_vpc()
subnets = self.autoscaler.get_subnets(vpc["VpcId"], zones)
target_group = self.loadbalancer.create_target_group(
    self.protocol, self.port, vpc["VpcId"]
)
self.loadbalancer.create_load_balancer(
    [subnet["SubnetId"] for subnet in subnets], target_group
)
self.autoscaler.attach_load_balancer_target_group(target_group)
print(f"Verifying access to the load balancer endpoint...")
lb_success = self.loadbalancer.verify_load_balancer_endpoint()
if not lb_success:
    print(
        "Couldn't connect to the load balancer, verifying that the port
is open..."
    )
)
```

```

        current_ip_address = requests.get(
            "http://checkip.amazonaws.com"
        ).text.strip()
        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:
            print(
                "For 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 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()
        if lb_success:
            print("Your load balancer is ready. You can access it by browsing to:
\n")
            print(f"\thttp://{self.loadbalancer.endpoint()}\n")

```

```
    else:
        print(
            "Couldn'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"
        )
        print(f"\thttp://{self.loadbalancer.endpoint()}\n")
    print("-" * 88)
    q.ask("Press Enter when you're ready to continue with the demo.")

def demo_choices(self):
    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:
        print("-" * 88)
        print(
            "\nSee the current state of the service by selecting one of the
following choices:\n"
        )
        choice = q.choose("\nWhich action would you like to take? ", actions)
        print("-" * 88)
        if choice == 0:
            print("Request:\n")
            print(f"GET http://{self.loadbalancer.endpoint()}")
            response = requests.get(f"http://{self.loadbalancer.endpoint()}")
            print("\nResponse:\n")
            print(f"{response.status_code}")
            if response.headers.get("content-type") == "application/json":
                pp(response.json())
        elif choice == 1:
            print("\nChecking the health of load balancer targets:\n")
            health = self.loadbalancer.check_target_health()
            for target in health:
                state = target["TargetHealth"]["State"]
                print(
                    f"\tTarget {target['Target']['Id']} on port
{target['Target']['Port']} is {state}"
                )
```



```
)
self.param_helper.put(self.param_helper.table, "this-is-not-a-table")
print(
    "\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."
)
self.demo_choices()

print(
    f"Instead of failing when the recommendation service fails, the web
service can return a static response.\n"
    f"While this is not a perfect solution, it presents the customer with
a somewhat better experience than failure.\n"
)
self.param_helper.put(self.param_helper.failure_response, "static")
print(
    f"\nNow, sending a GET request to the load balancer endpoint returns
a static response.\n"
    f"The service still reports as healthy because health checks are
still shallow.\n"
)
self.demo_choices()

print("Let's reinstate the recommendation service.\n")
self.param_helper.put(self.param_helper.table,
self.recommendation.table_name)
print(
    "\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"
)
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)
print(
```

```
        f"\nReplacing the profile for instance {bad_instance_id} with a
profile that contains\n"
        f"bad credentials...\n"
    )
    self.autoscaler.replace_instance_profile(
        bad_instance_id,
        self.autoscaler.bad_creds_profile_name,
        instance_profile["AssociationId"],
    )
    print(
        "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"
    )
    self.demo_choices()

    print(
        "\nLet's implement a deep health check. For this demo, a deep health
check tests whether\n"
        "the web service can access the DynamoDB table that it depends on for
recommendations. Note that\n"
        "the deep health check is only for ELB routing and not for Auto
Scaling instance health.\n"
        "This kind of deep health check is not recommended for Auto Scaling
instance health, because it\n"
        "risks accidental termination of all instances in the Auto Scaling
group when a dependent service fails.\n"
    )
    print(
        "By implementing deep health checks, the load balancer can detect
when one of the instances is failing\n"
        "and take that instance out of rotation.\n"
    )
    self.param_helper.put(self.param_helper.health_check, "deep")
    print(
        f"\nNow, checking target health indicates that the instance with bad
credentials ({bad_instance_id})\n"
        f"is unhealthy. Note that it might take a minute or two for the load
balancer to detect the unhealthy \n"
        f"instance. Sending a GET request to the load balancer endpoint
always returns a recommendation, because\n"
        "the load balancer takes unhealthy instances out of its rotation.\n"
    )
    self.demo_choices()
```



```
        print(
            "\nBecause the instances in this demo are controlled by an auto
scaler, the simplest way to fix an unhealthy\n"
            "instance is to terminate it and let the auto scaler start a new
instance to replace it.\n"
        )
        self.autoscaler.terminate_instance(bad_instance_id)
        print(
            "\nEven while the instance is terminating and the new instance is
starting, sending a GET\n"
            "request to the web service continues to get a successful
recommendation response because\n"
            "the load balancer routes requests to the healthy instances. After
the replacement instance\n"
            "starts and reports as healthy, it is included in the load balancing
rotation.\n"
            "\nNote that terminating and replacing an instance typically takes
several minutes, during which time you\n"
            "can see the changing health check status until the new instance is
running and healthy.\n"
        )
        self.demo_choices()

        print(
            "\nIf the recommendation service fails now, deep health checks mean
all instances report as unhealthy.\n"
        )
        self.param_helper.put(self.param_helper.table, "this-is-not-a-table")
        print(
            "\nWhen all instances are unhealthy, the load balancer continues to
route requests even to\n"
            "unhealthy instances, allowing them to fail open and return a static
response rather than fail\n"
            "closed and report failure to the customer."
        )
        self.demo_choices()
        self.param_helper.reset()

    def destroy(self):
        print(
            "This concludes the demo of how to build and manage a resilient
service.\n"
```

```
        "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 q.ask("Do you want to clean up all demo resources? (y/n) ",
q.is_yesno):
        self.loadbalancer.delete_load_balancer()
        self.loadbalancer.delete_target_group()
        self.autoscaler.delete_group()
        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,
        )
        self.recommendation.destroy()
    else:
        print(
            "Okay, 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."
        )

def main():
    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="../../../workflows/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()

    print("-" * 88)
```

```
print(
    "Welcome to the demonstration of How to Build and Manage a Resilient
    Service!"
)
print("-" * 88)

prefix = "doc-example-resilience"
recommendation = RecommendationService.from_client(
    "doc-example-recommendation-service"
)
autoscaler = AutoScaler.from_client(prefix)
loadbalancer = LoadBalancer.from_client(prefix)
param_helper = ParameterHelper.from_client(recommendation.table_name)
runner = Runner(
    args.resource_path, recommendation, autoscaler, loadbalancer,
    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()

print("-" * 88)
print("Thanks for watching!")
print("-" * 88)

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 AutoScaler:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """
```

```
def __init__(
    self,
    resource_prefix,
    inst_type,
    ami_param,
    autoscaling_client,
    ec2_client,
    ssm_client,
    iam_client,
):
    """
    :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
    self.launch_template_name = f"{resource_prefix}-template"
    self.group_name = f"{resource_prefix}-group"
    self.instance_policy_name = f"{resource_prefix}-pol"
    self.instance_role_name = f"{resource_prefix}-role"
    self.instance_profile_name = f"{resource_prefix}-prof"
    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"
    self.key_pair_name = f"{resource_prefix}-key-pair"

    @classmethod
    def from_client(cls, resource_prefix):
        """
        Creates this class from Boto3 clients.
```

```

    :param resource_prefix: The prefix for naming AWS resources that are
    created by this class.
    """
    as_client = boto3.client("autoscaling")
    ec2_client = boto3.client("ec2")
    ssm_client = boto3.client("ssm")
    iam_client = boto3.client("iam")
    return cls(
        resource_prefix,
        "t3.micro",
        "/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2",
        as_client,
        ec2_client,
        ssm_client,
        iam_client,
    )

    def create_instance_profile(
        self, policy_file, policy_name, role_name, profile_name,
        aws_managed_policies=()
    ):
        """
        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",
        }
    ],
}
with open(policy_file) as file:
    instance_policy_doc = file.read()

policy_arn = None
try:
    pol_response = self.iam_client.create_policy(
        PolicyName=policy_name, PolicyDocument=instance_policy_doc
    )
    policy_arn = pol_response["Policy"]["Arn"]
    log.info("Created policy with ARN %s.", policy_arn)
except ClientError as err:
    if err.response["Error"]["Code"] == "EntityAlreadyExists":
        log.info("Policy %s already exists, nothing to do.", policy_name)
        list_pol_response = self.iam_client.list_policies(Scope="Local")
        for pol in list_pol_response["Policies"]:
            if pol["PolicyName"] == policy_name:
                policy_arn = pol["Arn"]
                break
    if policy_arn is None:
        raise AutoScalerError(f"Couldn't create policy {policy_name}:
{err}")

try:
    self.iam_client.create_role(
        RoleName=role_name,
        AssumeRolePolicyDocument=json.dumps(assume_role_doc)
    )
    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("Created role %s and attached policy %s.", role_name,
policy_arn)
    except ClientError as err:
        if err.response["Error"]["Code"] == "EntityAlreadyExists":
            log.info("Role %s already exists, nothing to do.", role_name)
        else:
            raise AutoScalerError(f"Couldn't create role {role_name}: {err}")

    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
            )
        else:
            raise AutoScalerError(
                f"Couldn't create profile {profile_name} and attach it to
role\n"
                f"{role_name}: {err}")
    )
    return profile_arn

def get_instance_profile(self, instance_id):
    """
    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]}]
        )
    except ClientError as err:
        raise AutoScalerError(
            f"Couldn't get instance profile association for instance
{instance_id}: {err}"
        )
    else:
        return response["IamInstanceProfileAssociations"][0]

def replace_instance_profile(
    self, instance_id, new_instance_profile_name, profile_association_id
):
    """
    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 update.
    :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)
    inst_ready = False
    tries = 0
    while not inst_ready:
        if tries % 6 == 0:
            self.ec2_client.reboot_instances(InstanceIds=[instance_id])
            log.info(
                "Rebooting instance %s and waiting for it to be
ready.",
                instance_id,
            )
            tries += 1
            time.sleep(10)
            response = self.ssm_client.describe_instance_information()
            for info in response["InstanceInformationList"]:
                if info["InstanceId"] == instance_id:
                    inst_ready = True
            self.ssm_client.send_command(
                InstanceIds=[instance_id],
                DocumentName="AWS-RunShellScript",
                Parameters={"commands": ["cd / && sudo python3 server.py 80"]},
            )
            log.info("Restarted the Python web server on instance %s.",
instance_id)
        except ClientError as err:
            raise AutoScalerError(
                f"Couldn't replace instance profile for association
{profile_association_id}: {err}"
            )

    def delete_instance_profile(self, profile_name, role_name):
        """
        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:
        if err.response["Error"]["Code"] == "NoSuchEntity":
            log.info(
                "Instance profile %s doesn't exist, nothing to do.",
profile_name
            )
        else:
            raise AutoScalerError(
                f"Couldn't delete instance profile {profile_name} or detach "
                f"policies and delete role {role_name}: {err}"
            )

def create_key_pair(self, key_pair_name):
    """
    Creates a new key pair.

    :param key_pair_name: The name of the key pair to create.
    :return: The newly created key pair.
    """
    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:
        raise AutoScalerError(f"Couldn't create key pair {key_pair_name}:
{err}")

def delete_key_pair(self):
    """
    Deletes a key pair.

    :param key_pair_name: The name of the key pair to delete.
    """
    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:
        raise AutoScalerError(
            f"Couldn't delete key pair {self.key_pair_name}: {err}"
        )
    except FileNotFoundError:
        log.info("Key pair %s doesn't exist, nothing to do.",
self.key_pair_name)
    except PermissionError:
        log.info(
            "Inadequate permissions to delete key pair %s.",
self.key_pair_name
        )
    except Exception as err:
        raise AutoScalerError(
            f"Couldn't delete key pair {self.key_pair_name}: {err}"
        )

def create_template(self, server_startup_script_file, instance_policy_file):
    """
    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.
```

```
run
    :param server_startup_script_file: The path to a Bash script file that is
                                     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:
        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,
        )
        with open(server_startup_script_file) as file:
            start_server_script = file.read()
        ami_latest = self.ssm_client.get_parameter(Name=self.ami_param)
        ami_id = ami_latest["Parameter"]["Value"]
        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(
            "Created launch template %s for AMI %s on %s.",
            self.launch_template_name,
            ami_id,
            self.inst_type,
        )
    except ClientError as err:
        if (
            err.response["Error"]["Code"]
```

```
        == "InvalidLaunchTemplateName.AlreadyExistsException"
    ):
        log.info(
            "Launch template %s already exists, nothing to do.",
            self.launch_template_name,
        )
    else:
        raise AutoScalerError(
            f"Couldn't create launch template
{self.launch_template_name}: {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,
            )
        else:
            raise AutoScalerError(
                f"Couldn't delete launch template
{self.launch_template_name}: {err}."
            )

def get_availability_zones(self):
    """
```

```
    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"]]
    except ClientError as err:
        raise AutoScalerError(f"Couldn't get availability zones: {err}.")
    else:
        return zones

def create_group(self, group_size):
    """
    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.
    """
    zones = []
    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(
            "Created EC2 Auto Scaling group %s with availability zones %s.",
            self.launch_template_name,
            zones,
        )
    except ClientError as err:
        if err.response["Error"]["Code"] == "AlreadyExists":
            log.info(
```

```
        "EC2 Auto Scaling group %s already exists, nothing to do.",
        self.group_name,
    )
    else:
        raise AutoScalerError(
            f"Couldn't create EC2 Auto Scaling group {self.group_name}:
{err}"
        )
    return zones

def get_instances(self):
    """
    Gets data about the instances in the EC2 Auto Scaling group.

    :return: Data about the instances.
    """
    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"]
        ]
    except ClientError as err:
        raise AutoScalerError(
            f"Couldn't get instances for Auto Scaling group
{self.group_name}: {err}"
        )
    else:
        return instance_ids

def terminate_instance(self, instance_id):
    """
    Terminates and instances 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.
    """
    try:
        self.autoscaling_client.terminate_instance_in_auto_scaling_group(
```

```
        InstanceId=instance_id, ShouldDecrementDesiredCapacity=False
    )
    log.info("Terminated instance %s.", instance_id)
except ClientError as err:
    raise AutoScalerError(f"Couldn't terminate instance {instance_id}:
{err}")

def attach_load_balancer_target_group(self, lb_target_group):
    """
    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.

    :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:
        raise AutoScalerError(
            f"Couldn't attach load balancer target group
{lb_target_group['TargetGroupName']}\n"
            f"to auto scaling group {self.group_name}"
        )

def _try_terminate_instance(self, inst_id):
    stopping = False
    log.info(f"Stopping {inst_id}.")
    while not stopping:
        try:
            self.autoscaling_client.terminate_instance_in_auto_scaling_group(
                InstanceId=inst_id, ShouldDecrementDesiredCapacity=True
            )
```



```
        stopping = True
    except ClientError as err:
        if err.response["Error"]["Code"] == "ScalingActivityInProgress":
            log.info("Scaling activity in progress for %s. Waiting...",
inst_id)
                time.sleep(10)
            else:
                raise AutoScalerError(f"Couldn't stop instance {inst_id}:
{err}.")

    def _try_delete_group(self):
        """
        Tries to delete the EC2 Auto Scaling group. If the group is in use or in
progress,
the function waits and retries until the group is successfully deleted.
        """
        stopped = False
        while not stopped:
            try:
                self.autoscaling_client.delete_auto_scaling_group(
                    AutoScalingGroupName=self.group_name
                )
                stopped = True
                log.info("Deleted EC2 Auto Scaling group %s.", self.group_name)
            except ClientError as err:
                if (
                    err.response["Error"]["Code"] == "ResourceInUse"
                    or err.response["Error"]["Code"] ==
"ScalingActivityInProgress"
                ):
                    log.info(
                        "Some instances are still running. Waiting for them to
stop..."
                    )
                    time.sleep(10)
                else:
                    raise AutoScalerError(
                        f"Couldn't delete group {self.group_name}: {err}."
                    )

    def delete_group(self):
        """
        Terminates all instances in the group, deletes the EC2 Auto Scaling
group.
```

```
    """
    try:
        response = self.autoscaling_client.describe_auto_scaling_groups(
            AutoScalingGroupNames=[self.group_name]
        )
        groups = response.get("AutoScalingGroups", [])
        if len(groups) > 0:
            self.autoscaling_client.update_auto_scaling_group(
                AutoScalingGroupName=self.group_name, MinSize=0
            )
            instance_ids = [inst["InstanceId"] for inst in groups[0]
["Instances"]]
            for inst_id in instance_ids:
                self._try_terminate_instance(inst_id)
                self._try_delete_group()
            else:
                log.info("No groups found named %s, nothing to do.",
self.group_name)
        except ClientError as err:
            raise AutoScalerError(f"Couldn't delete group {self.group_name}:
{err}.")

    def get_default_vpc(self):
        """
        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:
            raise AutoScalerError(f"Couldn't get default VPC: {err}")
        else:
            return response["Vpcs"][0]

    def verify_inbound_port(self, vpc, port, ip_address):
        """
        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 specific 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("Found default security group %s.", sec_group["GroupId"])
```

```
for ip_perm in sec_group["IpPermissions"]:
```

```
    if ip_perm.get("FromPort", 0) == port:
```

```
        log.info("Found inbound rule: %s", 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(
```

```
                "The inbound rule does not appear to be open to  
either this computer's IP\n"
```

```
                "address of %s, to all IP addresses (0.0.0.0/0), or  
to a prefix list ID.",
```

```
                ip_address,
```

```
            )
```

```
        else:
```

```
            break
```

```
    except ClientError as err:
        raise AutoScalerError(
            f"Couldn't verify inbound rule for port {port} for VPC
{vpc['VpcId']}: {err}"
        )
    else:
        return sec_group, port_is_open

def open_inbound_port(self, sec_group_id, port, ip_address):
    """
    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:
        raise AutoScalerError(
            f"Couldn't authorize ingress to {sec_group_id} on port {port}
from {ip_address}: {err}"
        )

def get_subnets(self, vpc_id, zones):
    """
    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.
"""
try:
    response = self.ec2_client.describe_subnets(
        Filters=[
            {"Name": "vpc-id", "Values": [vpc_id]},
            {"Name": "availability-zone", "Values": zones},
            {"Name": "default-for-az", "Values": ["true"]},
        ]
    )
    subnets = response["Subnets"]
    log.info("Found %s subnets for the specified zones.", len(subnets))
except ClientError as err:
    raise AutoScalerError(f"Couldn't get subnets: {err}")
else:
    return subnets

```

Membuat kelas yang menggabungkan tindakan Penyeimbangan Beban Elastis.

```

class LoadBalancer:
    """Encapsulates Elastic Load Balancing (ELB) actions."""

    def __init__(self, target_group_name, load_balancer_name, elb_client):
        """
        :param target_group_name: The name of the target group associated with
        the load balancer.
        :param load_balancer_name: The name of the load balancer.
        :param elb_client: A Boto3 Elastic Load Balancing client.
        """
        self.target_group_name = target_group_name
        self.load_balancer_name = load_balancer_name
        self.elb_client = elb_client
        self._endpoint = None

    @classmethod

```

```
def from_client(cls, resource_prefix):
    """
    Creates this class from a Boto3 client.

    :param resource_prefix: The prefix to give to AWS resources created by
    this class.
    """
    elb_client = boto3.client("elbv2")
    return cls(f"{resource_prefix}-tg", f"{resource_prefix}-lb", elb_client)

def endpoint(self):
    """
    Gets the HTTP endpoint of the load balancer.

    :return: The endpoint.
    """
    if self._endpoint is None:
        try:
            response = self.elb_client.describe_load_balancers(
                Names=[self.load_balancer_name]
            )
            self._endpoint = response["LoadBalancers"][0]["DNSName"]
        except ClientError as err:
            raise LoadBalancerError(
                f"Couldn't get the endpoint for load balancer
                {self.load_balancer_name}: {err}"
            )
        return self._endpoint

def create_target_group(self, protocol, port, vpc_id):
    """
    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.

    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 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=self.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("Created load balancing target group %s.",
self.target_group_name)
except ClientError as err:
    raise LoadBalancerError(
        f"Couldn't create load balancing target group
{self.target_group_name}: {err}")
    )
else:
    return target_group

def delete_target_group(self):
    """
    Deletes the target group.
    """
    done = False
    while not done:
        try:
            response = self.elb_client.describe_target_groups(
                Names=[self.target_group_name]
            )
            tg_arn = response["TargetGroups"][0]["TargetGroupArn"]
            self.elb_client.delete_target_group(TargetGroupArn=tg_arn)
            log.info(
                "Deleted load balancing target group %s.",
self.target_group_name
```

```
        )
        done = True
    except ClientError as err:
        if err.response["Error"]["Code"] == "TargetGroupNotFound":
            log.info(
                "Load balancer target group %s not found, nothing to
do.",
                self.target_group_name,
            )
            done = True
        elif err.response["Error"]["Code"] == "ResourceInUse":
            log.info(
                "Target group not yet released from load balancer,
waiting..."
            )
            time.sleep(10)
        else:
            raise LoadBalancerError(
                f"Couldn't delete load balancing target group
{self.target_group_name}: {err}")
            )

    def create_load_balancer(self, subnet_ids, target_group):
        """
        Creates an Elastic Load Balancing load balancer that uses the specified
subnets
and forwards requests to the specified target group.

:param subnet_ids: A list of subnets to associate with the load balancer.
:param target_group: An existing target group that is added as a listener
to the
                load balancer.
:return: Data about the newly created load balancer.
        """
        try:
            response = self.elb_client.create_load_balancer(
                Name=self.load_balancer_name, Subnets=subnet_ids
            )
            load_balancer = response["LoadBalancers"][0]
            log.info("Created load balancer %s.", self.load_balancer_name)
            waiter = self.elb_client.get_waiter("load_balancer_available")
            log.info("Waiting for load balancer to be available...")
            waiter.wait(Names=[self.load_balancer_name])
```



```
log.info("Load balancer is available!")
self.elb_client.create_listener(
    LoadBalancerArn=load_balancer["LoadBalancerArn"],
    Protocol=target_group["Protocol"],
    Port=target_group["Port"],
    DefaultActions=[
        {
            "Type": "forward",
            "TargetGroupArn": target_group["TargetGroupArn"],
        }
    ],
)
log.info(
    "Created listener to forward traffic from load balancer %s to
target group %s.",
    self.load_balancer_name,
    target_group["TargetGroupName"],
)
except ClientError as err:
    raise LoadBalancerError(
        f"Failed to create load balancer {self.load_balancer_name}"
        f"and add a listener for target group
{target_group['TargetGroupName']}: {err}"
    )
else:
    self._endpoint = load_balancer["DNSName"]
    return load_balancer

def delete_load_balancer(self):
    """
    Deletes a load balancer.
    """
    try:
        response = self.elb_client.describe_load_balancers(
            Names=[self.load_balancer_name]
        )
        lb_arn = response["LoadBalancers"][0]["LoadBalancerArn"]
        self.elb_client.delete_load_balancer(LoadBalancerArn=lb_arn)
        log.info("Deleted load balancer %s.", self.load_balancer_name)
        waiter = self.elb_client.get_waiter("load_balancers_deleted")
        log.info("Waiting for load balancer to be deleted...")
        waiter.wait(Names=[self.load_balancer_name])
    except ClientError as err:
```

```
        if err.response["Error"]["Code"] == "LoadBalancerNotFound":
            log.info(
                "Load balancer %s does not exist, nothing to do.",
                self.load_balancer_name,
            )
        else:
            raise LoadBalancerError(
                f"Couldn't delete load balancer {self.load_balancer_name}:
{err}"
            )

    def verify_load_balancer_endpoint(self):
        """
        Verify this computer can successfully send a GET request to the load
        balancer endpoint.
        """
        success = False
        retries = 3
        while not success and retries > 0:
            try:
                lb_response = requests.get(f"http://{self.endpoint()}")
                log.info(
                    "Got response %s from load balancer endpoint.",
                    lb_response.status_code,
                )
                if lb_response.status_code == 200:
                    success = True
                else:
                    retries = 0
            except requests.exceptions.ConnectionError:
                log.info(
                    "Got connection error from load balancer endpoint,
retrying..."
                )
                retries -= 1
                time.sleep(10)
        return success

    def check_target_health(self):
        """
        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=[self.target_group_name]
        )
        health_response = self.elb_client.describe_target_health(
            TargetGroupArn=tg_response["TargetGroups"][0]["TargetGroupArn"]
        )
    except ClientError as err:
        raise LoadBalancerError(
            f"Couldn't check health of {self.target_group_name} targets:
{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, dynamodb_client):
        """
        :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

    @classmethod
    def from_client(cls, table_name):
        """
        Creates this class from a Boto3 client.
```

```
    :param table_name: The name of the DynamoDB recommendations table.
    """
    ddb_client = boto3.client("dynamodb")
    return cls(table_name, ddb_client)

    def create(self):
        """
        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.

        :return: Data about the newly created table.
        """
        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 do.", self.table_name)
            else:
                raise RecommendationServiceError(
                    self.table_name, f"ClientError when creating table: {err}."
                )
        else:
            return response
```

```
def populate(self, data_file):
    """
    Populates the recommendations table from a JSON file.

    :param data_file: The path to the data file.
    """
    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):
        """
        Deletes the recommendations table.
        """
        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 menggabungkan 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 = "doc-example-resilient-architecture-table"
    failure_response = "doc-example-resilient-architecture-failure-response"
    health_check = "doc-example-resilient-architecture-health-check"

    def __init__(self, table_name, ssm_client):
        """
        :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

    @classmethod
    def from_client(cls, table_name):
        ssm_client = boto3.client("ssm")
        return cls(table_name, ssm_client)

    def reset(self):
        """
        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, value):
        """
        Sets the value of a named Systems Manager parameter.
```

```
:param name: The name of the parameter.
:param value: The new value of the parameter.
"""
try:
    self.ssm_client.put_parameter(
        Name=name, Value=value, Overwrite=True, Type="String"
    )
    log.info("Setting demo parameter %s to '%s'.", name, value)
except ClientError as err:
    raise ParameterHelperError(
        f"Couldn't set parameter {name} to {value}: {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](#)
 - [DescribeInstanceProfileAssociations](#)
 - [DescribeInstances](#)
 - [DescribeLoadBalancers](#)
 - [DescribeSubnets](#)

- [DescribeTargetGroups](#)
- [DescribeTargetHealth](#)
- [DescribeVpcs](#)
- [RebootInstances](#)
- [ReplacelamInstanceProfileAssociation](#)
- [TerminateInstanceInAutoScalingGroup](#)
- [UpdateAutoScalingGroup](#)

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Buat grup IAM dan tambahkan pengguna ke grup menggunakan SDK AWS

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Buat grup dan berikan izin akses Amazon S3 penuh ke grup tersebut.
- Buat pengguna baru tanpa izin untuk mengakses Amazon S3.
- Tambahkan pengguna ke grup dan tunjukkan bahwa mereka sekarang memiliki izin untuk Amazon S3, lalu bersihkan sumber daya.

.NET

AWS SDK for .NET

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya 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>
    /// Add an existing IAM user to an existing IAM group.
    /// </summary>
    /// <param name="userName">The username of the user to add.</param>
    /// <param name="groupName">The name of the group to add the user to.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AddUserToGroupAsync(string userName, string
groupName)
    {
        var response = await _IAMService.AddUserToGroupAsync(new
AddUserToGroupRequest
        {
            GroupName = groupName,
            UserName = userName,
        });

        return response.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <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 group.
    /// </summary>
    /// <param name="groupName">The name to give the IAM group.</param>
    /// <returns>The IAM group that was created.</returns>
    public async Task<Group> CreateGroupAsync(string groupName)
    {
```

```
        var response = await _IAMService.CreateGroupAsync(new CreateGroupRequest
{ GroupName = groupName });
        return response.Group;
    }

    /// <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 group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteGroupAsync(string groupName)
    {
        var response = await _IAMService.DeleteGroupAsync(new DeleteGroupRequest
{ GroupName = groupName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// Delete an IAM policy associated with an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group associated with the
    /// policy.</param>
    /// <param name="policyName">The name of the policy to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteGroupPolicyAsync(string groupName, string
policyName)
    {
        var request = new DeleteGroupPolicyRequest()
        {
            GroupName = groupName,
            PolicyName = policyName,
        };
    }
```

```
        var response = await _IAMService.DeleteGroupPolicyAsync(request);
        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>
    /// Remove a user from an IAM group.
    /// </summary>
    /// <param name="userName">The username of the user to remove.</param>
    /// <param name="groupName">The name of the IAM group to remove the user
from.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> RemoveUserFromGroupAsync(string userName, string
groupName)
    {
        // Remove the user from the group.
        var removeUserRequest = new RemoveUserFromGroupRequest()
        {
            UserName = userName,
            GroupName = groupName,
        }
    }
}
```

```
};

var response = await
_IAMService.RemoveUserFromGroupAsync(removeUserRequest);
return response.HttpStatusCode == HttpStatusCode.OK;
}

/// <summary>
/// Add or update an inline policy document that is embedded in an IAM group.
/// </summary>
/// <param name="groupName">The name of the IAM group.</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> PutGroupPolicyAsync(string groupName, string
policyName, string policyDocument)
{
    var request = new PutGroupPolicyRequest
    {
        GroupName = groupName,
        PolicyName = policyName,
        PolicyDocument = policyDocument
    };

    var response = await _IAMService.PutGroupPolicyAsync(request);
    return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
}

/// <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 IAMGroups;

public class IAMGroups
{
    private static ILogger logger = null!;

    // Represents JSON code for AWS full access policy for Amazon Simple
    // Storage Service (Amazon S3).
    private const string S3FullAccessPolicyDocument = "{" +
        " \"Statement\" : [{" +
            " \"Action\" : [\"s3:*\"],\" +
            " \"Effect\" : \"Allow\",\" +
            " \"Resource\" : \"*\")\" +
        "}]\" +
    "};

    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<IAMGroups>();

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();

var groupUserName = configuration["GroupUserName"];
var groupName = configuration["GroupName"];
var groupPolicyName = configuration["GroupPolicyName"];
var groupBucketName = configuration["GroupBucketName"];

var wrapper = host.Services.GetRequiredService<IAMWrapper>();
var uiWrapper = host.Services.GetRequiredService<UIWrapper>();

uiWrapper.DisplayGroupsOverview();
uiWrapper.PressEnter();

// Create an IAM group.
uiWrapper.DisplayTitle("Create IAM group");
Console.WriteLine("Let's begin by creating a new IAM group.");
var group = await wrapper.CreateGroupAsync(groupName);

// Add an inline IAM policy to the group.
uiWrapper.DisplayTitle("Add policy to group");
Console.WriteLine("Add an inline policy to the group that allows members
to have full access to");
Console.WriteLine("Amazon Simple Storage Service (Amazon S3) buckets.");
```



```
await wrapper.PutGroupPolicyAsync(group.GroupName, groupPolicyName,
S3FullAccessPolicyDocument);

uiWrapper.PressEnter();

// Now create a new user.
uiWrapper.DisplayTitle("Create an IAM user");
Console.WriteLine("Now let's create a new IAM user.");
var groupUser = await wrapper.CreateUserAsync(groupUserName);

// Add the new user to the group.
uiWrapper.DisplayTitle("Add the user to the group");
Console.WriteLine("Adding the user to the group, which will give the user
the same permissions as the group.");
await wrapper.AddUserToGroupAsync(groupUser.UserName, group.GroupName);

Console.WriteLine($"User, {groupUser.UserName}, has been added to the
group, {group.GroupName}.");
uiWrapper.PressEnter();

Console.WriteLine("Now that we have created a user, and added the user to
the group, let's create an IAM access key.");

// Create access and secret keys for the user.
var accessKey = await wrapper.CreateAccessKeyAsync(groupUserName);
Console.WriteLine("Key created.");
uiWrapper.WaitABit(15, "Waiting for the access key to be ready for
use.");

uiWrapper.DisplayTitle("List buckets");
Console.WriteLine("To prove that the user has access to Amazon S3, list
the S3 buckets for the account.");

var s3Client = new AmazonS3Client(accessKey.AccessKeyId,
accessKey.SecretAccessKey);
var stsClient = new
AmazonSecurityTokenServiceClient(accessKey.AccessKeyId,
accessKey.SecretAccessKey);

var s3Wrapper = new S3Wrapper(s3Client, stsClient);

var buckets = await s3Wrapper.ListMyBucketsAsync();

if (buckets is not null)
```

```
{
    buckets.ForEach(bucket =>
    {
        Console.WriteLine($"{bucket.BucketName}\tcreated on:
{bucket.CreationDate}");
    });
}

// Show that the user also has write access to Amazon S3 by creating
// a new bucket.
uiWrapper.DisplayTitle("Create a bucket");
Console.WriteLine("Since group members have full access to Amazon S3,
let's create a bucket.");
var success = await s3Wrapper.PutBucketAsync(groupBucketName);

if (success)
{
    Console.WriteLine($"Successfully created the bucket:
{groupBucketName}.");
}

uiWrapper.PressEnter();

Console.WriteLine("Let's list the user's S3 buckets again to show the new
bucket.");

buckets = await s3Wrapper.ListMyBucketsAsync();

if (buckets is not null)
{
    buckets.ForEach(bucket =>
    {
        Console.WriteLine($"{bucket.BucketName}\tcreated on:
{bucket.CreationDate}");
    });
}

uiWrapper.PressEnter();

uiWrapper.DisplayTitle("Clean up resources");
Console.WriteLine("First delete the bucket we created.");
await s3Wrapper.DeleteBucketAsync(groupBucketName);
```

```
        Console.WriteLine($"Now remove the user, {groupUserName}, from the group,
{groupName}.");
        await wrapper.RemoveUserFromGroupAsync(groupUserName, groupName);

        Console.WriteLine("Delete the user's access key.");
        await wrapper.DeleteAccessKeyAsync(accessKey.AccessKeyId, groupUserName);

        // Now we can safely delete the user.
        Console.WriteLine("Now we can delete the user.");
        await wrapper.DeleteUserAsync(groupUserName);

        uiWrapper.PressEnter();

        Console.WriteLine("Now we will delete the IAM policy attached to the
group.");
        await wrapper.DeleteGroupPolicyAsync(groupName, groupPolicyName);

        Console.WriteLine("Now we delete the IAM group.");
        await wrapper.DeleteGroupAsync(groupName);

        uiWrapper.PressEnter();

        Console.WriteLine("The IAM groups demo has completed.");

        uiWrapper.PressEnter();
    }
}

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();
}
}
```

- Lihat detail API di topik-topik berikut dalam Referensi API AWS SDK for .NET .
 - [AddUserToGroup](#)
 - [AttachRolePolicy](#)
 - [CreateAccessKey](#)
 - [CreateGroup](#)
 - [CreatePolicy](#)
 - [CreateRole](#)
 - [CreateUser](#)
 - [DeleteAccessKey](#)
 - [DeleteGroup](#)
 - [DeleteGroupPolicy](#)
 - [DeleteUser](#)

- [PutGroupPolicy](#)
- [RemoveUserFromGroup](#)

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Buat pengguna IAM dan ambil peran dengan AWS STS menggunakan SDK AWS

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 kredensial sementara, lalu bersihkan sumber daya.

.NET

AWS SDK for .NET

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya 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>
    /// Add an existing IAM user to an existing IAM group.
    /// </summary>
    /// <param name="userName">The username of the user to add.</param>
    /// <param name="groupName">The name of the group to add the user to.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AddUserToGroupAsync(string userName, string
groupName)
    {
        var response = await _IAMService.AddUserToGroupAsync(new
AddUserToGroupRequest
        {
            GroupName = groupName,
            UserName = userName,
        });

        return response.HttpStatusCode == HttpStatusCode.OK;
    }
}
```

```
    /// <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 group.
    /// </summary>
    /// <param name="groupName">The name to give the IAM group.</param>
```

```
    /// <returns>The IAM group that was created.</returns>
    public async Task<Group> CreateGroupAsync(string groupName)
    {
        var response = await _IAMService.CreateGroupAsync(new CreateGroupRequest
{ GroupName = groupName });
        return response.Group;
    }

    /// <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 group.
/// </summary>
/// <param name="groupName">The name of the IAM group to delete.</param>
/// <returns>A Boolean value indicating the success of the action.</returns>
public async Task<bool> DeleteGroupAsync(string groupName)
{
    var response = await _IAMService.DeleteGroupAsync(new DeleteGroupRequest
{ GroupName = groupName });
    return response.HttpStatusCode == HttpStatusCode.OK;
}

/// <summary>
/// Delete an IAM policy associated with an IAM group.
/// </summary>
/// <param name="groupName">The name of the IAM group associated with the
/// policy.</param>
/// <param name="policyName">The name of the policy to delete.</param>
/// <returns>A Boolean value indicating the success of the action.</returns>
public async Task<bool> DeleteGroupPolicyAsync(string groupName, string
policyName)
{
    var request = new DeleteGroupPolicyRequest()
    {
        GroupName = groupName,
```

```
        PolicyName = policyName,
    };

    var response = await _IAMService.DeleteGroupPolicyAsync(request);
    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>
/// Remove a user from an IAM group.
/// </summary>
/// <param name="userName">The username of the user to remove.</param>
/// <param name="groupName">The name of the IAM group to remove the user
from.</param>
/// <returns>A Boolean value indicating the success of the action.</returns>
public async Task<bool> RemoveUserFromGroupAsync(string userName, string
groupName)
{
    // Remove the user from the group.
    var removeUserRequest = new RemoveUserFromGroupRequest()
```

```
    {
        UserName = userName,
        GroupName = groupName,
    };

    var response = await
_IAMService.RemoveUserFromGroupAsync(removeUserRequest);
    return response.HttpStatusCode == HttpStatusCode.OK;
}

/// <summary>
/// Add or update an inline policy document that is embedded in an IAM group.
/// </summary>
/// <param name="groupName">The name of the IAM group.</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> PutGroupPolicyAsync(string groupName, string
policyName, string policyDocument)
{
    var request = new PutGroupPolicyRequest
    {
        GroupName = groupName,
        PolicyName = policyName,
        PolicyDocument = policyDocument
    };

    var response = await _IAMService.PutGroupPolicyAsync(request);
    return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
}

/// <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.WriteLine("\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();  
  }  
}
```

- Lihat detail API di topik-topik berikut dalam Referensi API AWS SDK for .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
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 getopt command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user"
    echo "Creates an WS 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 getopt 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 getopt "u:f:h" option; do
        case "${option}" in
            u) user_name="${OPTARG}" ;;
            f) file_name="${OPTARG}" ;;
        esac
    done
```

```
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 getopt 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 getopt "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
}

```

```
    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 getopt 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 getopt "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
    fi
}
```

```

    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 role.
#
# 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 getopt 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 role.
#
# 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 getopt 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 getopt "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 getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function iam_delete_policy"
        echo "Deletes an WS Identity and Access Management (IAM) policy"
        echo "  -n policy_arn -- The name of the IAM policy arn."
        echo ""
    }

    # Retrieve the calling parameters.
    while getopt "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 getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function iam_delete_role"
        echo "Deletes an WS Identity and Access Management (IAM) role"
        echo "  -n role_name -- The name of the IAM role."
        echo ""
    }

    # Retrieve the calling parameters.
    while getopt "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 getopt command in a function.

    # bashsupport disable=BP5008
    function usage() {
        echo "function iam_delete_access_key"
        echo "Deletes an WS 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 getopt "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."
    fi
}
```

```

    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 getopt command in a function.  
  
    # bashsupport disable=BP5008  
    function usage() {  
        echo "function iam_delete_user"  
        echo "Deletes an WS 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 getopt "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 for C++

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya 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::STSCliient 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;
}
```

- Lihat detail API di topik-topik berikut dalam Referensi API AWS SDK for C++ .
 - [AttachRolePolicy](#)
 - [CreateAccessKey](#)
 - [CreatePolicy](#)
 - [CreateRole](#)
 - [CreateUser](#)
 - [DeleteAccessKey](#)
 - [DeletePolicy](#)
 - [DeleteRole](#)
 - [DeleteUser](#)
 - [DeleteUserPolicy](#)
 - [DetachRolePolicy](#)

- [PutUserPolicy](#)

Go

SDK for Go V2

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

Jalankan skenario interaktif di penggugah/prompt perintah.

```
// 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() {
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()
accessKey := scenario.CreateAccessKey(user)
role := scenario.CreateRoleAndPolicies(user)
```

```
noPermsConfig := scenario.ListBucketsWithoutPermissions(accessKey)
scenario.ListBucketsWithAssumedRole(noPermsConfig, role)
scenario.Cleanup(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() *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(userName)
    if err != nil {
        panic(err)
    }
    if user == nil {
        user, err = scenario.userWrapper.CreateUser(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(user *types.User)
*types.AccessKey {
    accessKey, err := scenario.userWrapper.CreateAccessKeyPair(*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(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(scenario.helper.GetName(), *user.Arn)
    if err != nil {panic(err)}
    log.Printf("Created role %v.\n", *listBucketsRole.RoleName)
    listBucketsPolicy, err := scenario.policyWrapper.CreatePolicy(
    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(*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(*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(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(context.TODO(),
 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(context.TODO(), &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(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(context.TODO(),
    &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(context.TODO(),
    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(context.TODO(), &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(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(*role.RoleName)
        if err != nil {panic(err)}
        for _, policy := range policies {
            err = scenario.roleWrapper.DetachRolePolicy(*role.RoleName,
                *policy.PolicyArn)
            if err != nil {panic(err)}
            err = scenario.policyWrapper.DeletePolicy(*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(*role.RoleName)
        if err != nil {panic(err)}
        log.Printf("Deleted role %v.\n", *role.RoleName)

        userPols, err := scenario.userWrapper.ListUserPolicies(*user.UserName)
        if err != nil {panic(err)}
        for _, userPol := range userPols {
            err = scenario.userWrapper.DeleteUserPolicy(*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(*user.UserName)
        if err != nil {panic(err)}
        for _, key := range keys {
            err = scenario.userWrapper.DeleteAccessKey(*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(*user.UserName)
    if err != nil {panic(err)}
    log.Printf("Deleted user %v.\n", *user.UserName)
    log.Println(strings.Repeat("-", 88))
}
}

```

Tentukan struct yang membungkus tindakan akun.

```

// 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() (*types.PasswordPolicy,
error) {
    var pwPolicy *types.PasswordPolicy
    result, err := wrapper.IamClient.GetAccountPasswordPolicy(context.TODO(),
&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() ([]types.SAMLProviderListEntry,
error) {
    var providers []types.SAMLProviderListEntry
    result, err := wrapper.IamClient.ListSAMLProviders(context.TODO(),
&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.

```
// 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"`
}

// 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(maxPolicies int32) ([]types.Policy,
error) {
    var policies []types.Policy
    result, err := wrapper.IamClient.ListPolicies(context.TODO(),
&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
}

// 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(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(context.TODO(),
&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(policyArn string) (*types.Policy, error) {
    var policy *types.Policy
    result, err := wrapper.IamClient.GetPolicy(context.TODO(), &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(policyArn string) error {
    _, err := wrapper.IamClient.DeletePolicy(context.TODO(), &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.

```
// 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(maxRoles int32) ([]types.Role, error) {
    var roles []types.Role
    result, err := wrapper.IamClient.ListRoles(context.TODO(),
        &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(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(context.TODO(),
&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(roleName string) (*types.Role, error) {
    var role *types.Role
    result, err := wrapper.IamClient.GetRole(context.TODO(),
&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(serviceName string,
description string) (*types.Role, error) {
    var role *types.Role
    result, err := wrapper.IamClient.CreateServiceLinkedRole(context.TODO(),
&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(roleName string) error {
    _, err := wrapper.IamClient.DeleteServiceLinkedRole(context.TODO(),
&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(policyArn string, roleName string)
error {
    _, err := wrapper.IamClient.AttachRolePolicy(context.TODO(),
&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(roleName string)
([]types.AttachedPolicy, error) {
    var policies []types.AttachedPolicy
    result, err := wrapper.IamClient.ListAttachedRolePolicies(context.TODO(),
&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(roleName string, policyArn string)
error {
    _, err := wrapper.IamClient.DetachRolePolicy(context.TODO(),
&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(roleName string) ([]string, error) {
    var policies []string
    result, err := wrapper.IamClient.ListRolePolicies(context.TODO(),
&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(roleName string) error {
  _, err := wrapper.IamClient.DeleteRole(context.TODO(), &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.

```
// 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(maxUsers int32) ([]types.User, error) {
  var users []types.User
  result, err := wrapper.IamClient.ListUsers(context.TODO(), &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(userName string) (*types.User, error) {
  var user *types.User
  result, err := wrapper.IamClient.GetUser(context.TODO(), &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(userName string) (*types.User, error) {
  var user *types.User
  result, err := wrapper.IamClient.CreateUser(context.TODO(),
  &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(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(context.TODO(),
        &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(userName string) ([]string, error) {
    var policies []string
    result, err := wrapper.IamClient.ListUserPolicies(context.TODO(),
        &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(userName string, policyName string)
    error {
    _, err := wrapper.IamClient.DeleteUserPolicy(context.TODO(),
        &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(userName string) error {
    _, err := wrapper.IamClient.DeleteUser(context.TODO(), &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(userName string)
(*types.AccessKey, error) {
    var key *types.AccessKey
    result, err := wrapper.IamClient.CreateAccessKey(context.TODO(),
&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(userName string, keyId string) error {
    _, err := wrapper.IamClient.DeleteAccessKey(context.TODO(),
&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(userName string)
([]types.AccessKeyMetadata, error) {
    var keys []types.AccessKeyMetadata
    result, err := wrapper.IamClient.ListAccessKeys(context.TODO(),
    &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
}
```

- Lihat detail API di topik-topik berikut dalam Referensi API AWS SDK for Go .
 - [AttachRolePolicy](#)
 - [CreateAccessKey](#)
 - [CreatePolicy](#)
 - [CreateRole](#)
 - [CreateUser](#)
 - [DeleteAccessKey](#)
 - [DeletePolicy](#)
 - [DeleteRole](#)
 - [DeleteUser](#)
 - [DeleteUserPolicy](#)
 - [DetachRolePolicy](#)
 - [PutUserPolicy](#)

Java

SDK for 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);
    }
}
}
```

- Lihat detail API di topik-topik berikut dalam 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 kredensial sementara untuk membuat daftar bucket untuk akun.

```
import {
  CreateUserCommand,
  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";

// Set the parameters.
const iamClient = new IAMClient({});
const userName = "test_name";
const policyName = "test_policy";
const roleName = "test_role";

export const main = async () => {
  // Create a user. The user has no permissions by default.
  const { User } = await iamClient.send(
    new CreateUserCommand({ Username: userName }),
  );

  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: 60 }, () =>
  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"));
};
```

- Lihat detail API di topik-topik berikut dalam Referensi API AWS SDK for JavaScript .
 - [AttachRolePolicy](#)
 - [CreateAccessKey](#)
 - [CreatePolicy](#)
 - [CreateRole](#)
 - [CreateUser](#)
 - [DeleteAccessKey](#)
 - [DeletePolicy](#)
 - [DeleteRole](#)
 - [DeleteUser](#)
 - [DeleteUserPolicy](#)
 - [DetachRolePolicy](#)
 - [PutUserPolicy](#)

Kotlin

SDK for 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: String = "{" +
        "  \"Version\": \"2012-10-17\", " +
        "  \"Statement\": [" +
        "    {" +
        "      \"Effect\": \"Allow\", " +
        "      \"Action\": [" +
        "        \"s3:*\" " +
        "      ], " +
        "      \"Resource\": \"*\\" " +
        "    } " +
        "  ] " +
        "}"

    val request = CreatePolicyRequest {
        policyName = policyNameVal
        policyDocument = policyDocumentValue
    }

    IamClient { 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 { 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 {
    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 {
    credentialsProvider = staticCredentials
    region = "us-east-1"
  }
}
```

```
    }

    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 { 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 { 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)
}
```

- Lihat detail API di topik-topik berikut dalam Referensi API AWS SDK For Kotlin.
 - [AttachRolePolicy](#)
 - [CreateAccessKey](#)
 - [CreatePolicy](#)
 - [CreateRole](#)
 - [CreateUser](#)
 - [DeleteAccessKey](#)
 - [DeletePolicy](#)
 - [DeleteRole](#)
 - [DeleteUser](#)
 - [DeleteUserPolicy](#)
 - [DetachRolePolicy](#)
 - [PutUserPolicy](#)

PHP

SDK for PHP

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya 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";
```

- Lihat detail API di topik-topik berikut dalam Referensi API AWS SDK for PHP .
 - [AttachRolePolicy](#)
 - [CreateAccessKey](#)
 - [CreatePolicy](#)
 - [CreateRole](#)
 - [CreateUser](#)
 - [DeleteAccessKey](#)
 - [DeletePolicy](#)
 - [DeleteRole](#)
 - [DeleteUser](#)
 - [DeleteUserPolicy](#)
 - [DetachRolePolicy](#)

- [PutUserPolicy](#)

Python

SDK for 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 kredensial 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(
        why: "
        f"Couldn't create a policy and attach it to role {role.name}. Here's
        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()
```

- Lihat detail API di topik-topik berikut dalam Referensi API AWS SDK for Python (Boto3).
 - [AttachRolePolicy](#)
 - [CreateAccessKey](#)

- [CreatePolicy](#)
- [CreateRole](#)
- [CreateUser](#)
- [DeleteAccessKey](#)
- [DeletePolicy](#)
- [DeleteRole](#)
- [DeleteUser](#)
- [DeleteUserPolicy](#)
- [DetachRolePolicy](#)
- [PutUserPolicy](#)

Ruby

SDK for 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 kredensial 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
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!")
end
```



```
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__
```

- Lihat detail API di topik-topik berikut dalam Referensi API AWS SDK for 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, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Buat pengguna IAM read-only dan read-write menggunakan SDK AWS

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 kredensial 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:::my-bucket/*' to allow actions on all
        objects
        in the bucket named 'my-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!")
```

- Lihat detail API di topik-topik berikut dalam Referensi API AWS SDK for Python (Boto3).
 - [AttachUserPolicy](#)
 - [CreateAccessKey](#)
 - [CreatePolicy](#)
 - [CreateUser](#)
 - [DeleteAccessKey](#)
 - [DeletePolicy](#)
 - [DeleteUser](#)
 - [DetachUserPolicy](#)
 - [ListUsers](#)
 - [UpdateUser](#)

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Mengelola kunci akses IAM menggunakan SDK AWS

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!")
```

- Lihat detail API di topik-topik berikut dalam Referensi API AWS SDK for Python (Boto3).
 - [CreateAccessKey](#)
 - [DeleteAccessKey](#)
 - [GetAccessKeyLastUsed](#)
 - [ListAccessKeys](#)
 - [UpdateAccessKey](#)

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Mengelola kebijakan IAM menggunakan SDK AWS

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:::my-bucket/*' to allow actions on all
objects
                        in the bucket named 'my-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(
so "
                "Default version %s found for %s, but no previous version exists,
                "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:::made-up-bucket-name"
    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!")
```

- Lihat detail API di topik-topik berikut dalam Referensi API AWS SDK for Python (Boto3).
 - [CreatePolicy](#)
 - [CreatePolicyVersion](#)
 - [DeletePolicy](#)
 - [DeletePolicyVersion](#)
 - [GetPolicyVersion](#)
 - [ListPolicies](#)
 - [ListPolicyVersions](#)
 - [SetDefaultPolicyVersion](#)

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Mengelola peran IAM menggunakan SDK AWS

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!")
```

- Lihat detail API di topik-topik berikut dalam Referensi API AWS SDK for Python (Boto3).
 - [AttachRolePolicy](#)
 - [CreateRole](#)
 - [DeleteRole](#)
 - [DetachRolePolicy](#)

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Mengelola akun IAM Anda menggunakan SDK AWS

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Dapatkan dan perbarui alias akun.
- Hasilkan laporan pengguna dan kredensial.
- 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.")
```

- Lihat detail API di topik-topik berikut dalam Referensi API AWS SDK for Python (Boto3).
 - [CreateAccountAlias](#)
 - [DeleteAccountAlias](#)
 - [GenerateCredentialReport](#)
 - [GetAccountAuthorizationDetails](#)
 - [GetAccountSummary](#)
 - [GetCredentialReport](#)
 - [ListAccountAliases](#)

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Kembalikan versi kebijakan IAM menggunakan SDK AWS

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, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Bekerja dengan API Pembuat Kebijakan IAM menggunakan SDK AWS

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 for 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")

.addAction("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:::BUCKETNAME/*")

        .addResource("arn:aws:s3:::BUCKETNAME")
            .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::DOC-
EXAMPLE-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 fileIamPolicy.

```

    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 fileIamPolicy.

```
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](#).
- Lihat detail API di topik-topik berikut dalam Referensi API AWS SDK for Java 2.x .
 - [CreatePolicy](#)
 - [GetPolicy](#)
 - [GetPolicyVersion](#)

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Contoh kode untuk AWS STS menggunakan 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. Meskipun tindakan menunjukkan cara memanggil setiap fungsi layanan, Anda dapat melihat tindakan dalam konteks pada skenario yang terkait dan contoh lintas layanan.

Skenario adalah contoh kode yang menunjukkan cara untuk menyelesaikan tugas tertentu dengan memanggil beberapa fungsi dalam layanan yang sama.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Contoh kode

- [Tindakan untuk AWS STS menggunakan AWS SDK](#)
 - [Gunakan AssumeRole dengan AWS SDK atau alat baris perintah](#)
 - [Gunakan AssumeRoleWithWebIdentity dengan AWS SDK atau alat baris perintah](#)
 - [Gunakan DecodeAuthorizationMessage dengan AWS SDK atau alat baris perintah](#)

- [Gunakan GetFederationToken dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetSessionToken dengan AWS SDK atau alat baris perintah](#)
- [Skenario untuk AWS STS menggunakan AWS SDK](#)
 - [Asumsikan peran IAM yang membutuhkan token MFA AWS STS dengan menggunakan SDK AWS](#)
 - [Membangun URL dengan AWS STS untuk pengguna federasi menggunakan SDK AWS](#)
 - [Dapatkan token sesi yang memerlukan token MFA dengan AWS STS menggunakan SDK AWS](#)

Tindakan untuk AWS STS menggunakan AWS SDK

Contoh kode berikut menunjukkan cara melakukan AWS STS tindakan individual dengan AWS SDK. Kutipan ini memanggil AWS STS API dan merupakan kutipan kode dari program yang lebih besar yang harus dijalankan dalam konteks. Setiap contoh menyertakan tautan ke GitHub, di mana Anda dapat menemukan instruksi untuk mengatur dan menjalankan kode.

Contoh berikut hanya mencakup tindakan yang paling umum digunakan. Untuk daftar lengkapnya, lihat [Referensi API AWS Security Token Service \(AWS STS\)](#).

Contoh-contoh

- [Gunakan AssumeRole dengan AWS SDK atau alat baris perintah](#)
- [Gunakan AssumeRoleWithWebIdentity dengan AWS SDK atau alat baris perintah](#)
- [Gunakan DecodeAuthorizationMessage dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetFederationToken dengan AWS SDK atau alat baris perintah](#)
- [Gunakan GetSessionToken dengan AWS SDK atau alat baris perintah](#)

Gunakan **AssumeRole** dengan AWS SDK atau alat baris perintah

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

AWS SDK for .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>

        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);

        // 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 [AssumeRole](#) di Referensi AWS SDK for .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 getopt 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 getopt 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 [AssumeRole](#) di Referensi AWS CLI Perintah.

C++

SDK for 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 [AssumeRole](#) di Referensi AWS SDK for 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": "ARO3XFRBF535PLBIFPI4:s3-access-example",
    "Arn": "arn:aws:sts::123456789012:assumed-role/xaccounts3access/s3-
access-example"
  },
  "Credentials": {
```

```

    "SecretAccessKey": "9drTJvcXLB89EXAMPLELB8923FB892xMFI",
    "SessionToken": "AQoXdzELDDY//////////
wEaoAK1wvxJY12r2IrDFT2IvAzTCn3zHoZ7YNtpiQLF0MqZye/
qwjzP2iEXAMPLEbw/m3hsj8VBTkPORGvr9jM5sgP+w9IZWZnU+LWhmg
+a5fDi2oTGUYcdg9uexQ4mtCHIHfi4citgqZTgco40Yqr4lIlo4V2b2Dyauk0eYFNebHtY1FVgAUj
+7Indz3LU0aTWk1WKIjHmMCIoTkyYp/k7kUG7moeEYKSitwQIi6Gjn+nyzM
+PtoA3685ixzv0R7i5rjQi0YE0lf1oeie3bDiNHncmzosRM6SFIPzSvp6h/32xQuZsjcypmwsPSDtTPYcs0+YN/8B
IcrxSpnWEXAMPLEXSDFTAQAM6Dl9zR0tXoybnlrZIwMLlMi1Kcgo50ytwU=",
    "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 kredensial untuk Anda. Untuk informasi selengkapnya, lihat [Menggunakan peran IAM di AWS CLI](#) di Panduan Pengguna AWS CLI.

- Untuk detail API, lihat [AssumeRole](#) di Referensi AWS CLI Perintah.

Java

SDK for 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, rn: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 [AssumeRole](#) di 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 [AssumeRole](#) di Referensi AWS SDK for 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 [AssumeRole](#) di Referensi AWS SDK for JavaScript API.

PowerShell

Alat untuk PowerShell

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. Kredensial 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 kredensial 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 [AssumeRole](#) di Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK for 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 kredensial 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 [AssumeRole](#) di AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK for 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 [AssumeRole](#) di Referensi AWS SDK for 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 [AssumeRole](#) referensi AWS SDK for Rust API.

Swift

SDK for Swift

Note

Ini adalah dokumentasi prarilis untuk SDK dalam rilis pratinjau. Dokumentasi ini dapat berubah.

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
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 {
        throw error
    }
}
```

- Untuk detail API, lihat referensi [AssumeRole AWSSDK](#) untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **AssumeRoleWithWebIdentity** dengan AWS SDK atau alat baris perintah

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
%7CIQEBLjAsAhRFiXuWpUXuRvQ9PZL3GMFcYevydwIUFAHZwXZXXXXXXXXXJnrulxKDHwy87oGKPznh0D6bEQZTSCz
CrKqjG7nPBjNIL016GGvuS5gSvPRUxWES3VYfm1w17WTI7jn-Pcb6M-
buCgHhF0zTQxod27L9Cqn0Lio7N3gZAGpsp6n1-
AJB0CJckcyXe2c6uD0sr0JeZlKUm2eTDVMf8IehDVI0r1Q0nTV6KzzAI30Y87Vd_cVMQ"
```

Output:

```
{
  "SubjectFromWebIdentityToken": "amzn1.account.AF6RH07KZU5XRVQJGXX6HB56KR2A"
  "Audience": "client.5498841531868486423.1548@apps.example.com",
  "AssumedRoleUser": {
```

```

    "Arn": "arn:aws:sts::123456789012:assumed-role/FederatedWebIdentityRole/
app1",
    "AssumedRoleId": "AROACLKWSQRAOEXAMPLE:app1"
  }
  "Credentials": {
    "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxrFiCYzEXAMPLEKEY",
    "SessionToken": "AQoEXAMPLEH4aoAH0gNCAPyJxz4B1CFFxWNE10PTgk5TthT
+FvwqnKwRc0IfirRh3c/LTo6UDdyJw00vEVPvLXCrrrUtdnniCEXAMPLE/
IvU1dYUg2RVAJBanLiHb4IgRmpRV3zrkuWJ0gQs8IZZaIv2BXIa2R40lgkBN9bkUDNCJiBeb/
AXlzBBko7b15fjrBs2+cTQtpZ3CYWFXG8C5zqx37wn0E49mRl/+0tkIKG07fAE",
    "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 [AssumeRoleWithWebIdentity](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Mengembalikan kumpulan kredensial 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 dalam izin yang terkait dengan peran). 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 [AssumeRoleWithWebIdentity](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan `DecodeAuthorizationMessage` dengan AWS SDK atau alat baris perintah

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-
eTA6i6Dr1UhhPQrLWB_1Ab15pAKx19mPDLexYcGBreyIKQC1BGBIpBKr3dFDkwqe07e2NMk5j_hmzAiChJN-8oy3
0jau7BMj0TWw0tHPHV_Zaz87yENdipr745EjQwRd5LaoL3vN8_5ZfA9UiBMKDgVh1gjqZJFUiQoubv78V1RbHNYnK
p0u3FZjwYStfvTb3GHs3-6rLribG09jZ0ktkfE6vqx1FzLyeDr4P2ihC1wty9tArCvvGzIAUNmARQJ2VWVPxioqgo
JWP5pwe_mAyqh0NLw-r1S56YC_90onj9A80sNrHlI-
tIiNd7tgNTYzDuPQYD2FMDBnp82V9eVmYGtPp5NIeSpuf3f0HanFuBZgENxZQZ2d1H3xJGMTtYayzZrRXjiq_SfX9
FaoPIb8LmmKVBLpIB0iFhU9sEHPqKHVPi6jdxXqKaZaFGvYVmV0iuQdNQKuyk0p067P0FrZECLjj0tNPBOZCcuEKE
```

Output:

```
{
  "DecodedMessage": "{\"allowed\":false,\"explicitDeny\":true,
  \"matchedStatements\":{\"items\":[{\"statementId\":\"VisualEditor0\",\"effect
  \":\"DENY\",\"principals\":{\"items\":[{\"value\":\"ARO123456789EXAMPLE
  \"}]},\"principalGroups\":{\"items\":[]},\"actions\":{\"items\":[{\"value
  \":\"ec2:RunInstances\"}]},\"resources\":{\"items\":[{\"value\":\"*
  \"}]},\"conditions\":{\"items\":[]}]},\"failures\":{\"items\":[]},
  \"context\":{\"principal\":{\"id\":\"ARO123456789EXAMPLE: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:efsOptimized\", \"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\": \"efs\"}]}}], {\"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 [DecodeAuthorizationMessage](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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 [DecodeAuthorizationMessage](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **GetFederationToken** dengan AWS SDK atau alat baris perintah

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/bPxrFfiCYEXAMPLEKEY",
    "SessionToken": "EXAMPLEpZ21uX2VjEGoaCXVzLXdlc3QtMiJIMEYCIQC/
W9pL5ArQyDD5JwFL3/h5+WGopQ24GEXweNctwhi9sgIhAMkg
+MZE35iWM8s4r5Lr25f9rSTVPFH98G42QQuWMTfKq0DCOP////////
wEQAxoMNDUy0TI1MTcwNTA3Igxuy3A0puuoLsk3MJwqgQPg8Q0d9HuoClUxq26wnc/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 [GetFederationToken](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

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-STS FederationToken -Name "Bob" -Policy "...JSON policy..." -DurationInSeconds 3600
```

- Untuk detail API, lihat [GetFederationToken](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Gunakan **GetSessionToken** dengan AWS SDK atau alat baris perintah

Contoh kode berikut menunjukkan cara menggunakan `GetSessionToken`.

Contoh-contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan di dalam konteks. Anda dapat melihat tindakan ini dalam konteks pada 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": "AQoEXAMPLEH4aoAH0gNCAPyJxz4B1CFFxWNE10PTgk5TthT  
+FvwqnKwRc0IfrrRh3c/LTo6UDdyJw00vEVPvLXCrrrUtdnniCEXAMPLE/  
IvU1dYUg2RVAJBanLiHb4IgrmpRV3zrkuWJ0gQs8IZZaIv2BXIa2R40lgkBN9bkUDNCJiBeb/  
AX1zBBko7b15fjrBs2+cTQtpZ3CYWFXG8C5zqx37wn0E49mR1/+0tkIKG07fAE",  
    "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 [GetSessionToken](#) di Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Mengembalikan sebuah **Amazon.RuntimeAWSCredentials** instance yang berisi kredensial sementara yang valid untuk jangka waktu tertentu. Kredensial yang digunakan untuk meminta kredensial sementara disimpulkan dari default shell saat ini. Untuk menentukan kredensial 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 'myprofile' 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 [GetSessionToken](#) di Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK for 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 [GetSessionToken](#) di AWS SDK for Python (Boto3) Referensi API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Skenario untuk AWS STS menggunakan 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 dalamnya AWS STS. Setiap skenario menyertakan tautan ke GitHub, di mana Anda dapat menemukan petunjuk tentang cara mengatur dan menjalankan kode.

Contoh-contoh

- [Asumsikan peran IAM yang membutuhkan token MFA AWS STS dengan menggunakan SDK AWS](#)
- [Membangun URL dengan AWS STS untuk pengguna federasi menggunakan SDK AWS](#)
- [Dapatkan token sesi yang memerlukan token MFA dengan AWS STS menggunakan SDK AWS](#)

Asumsikan peran IAM yang membutuhkan token MFA AWS STS dengan menggunakan SDK AWS

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 kredensial MFA disediakan.
- Daftarkan perangkat MFA untuk pengguna.
- Asumsikan peran dan gunakan kredensial 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 [AssumeRole](#) di AWS SDK for Python (Boto3) Referensi API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Membangun URL dengan AWS STS untuk pengguna federasi menggunakan SDK AWS

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 kredensial 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 [AssumeRole](#) di AWS SDK for Python (Boto3) Referensi API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Dapatkan token sesi yang memerlukan token MFA dengan AWS STS menggunakan SDK AWS

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 kredensial MFA disediakan.
- Daftarkan perangkat MFA untuk pengguna.
- Berikan kredensial MFA untuk mendapatkan token sesi dan gunakan kredensial 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 kredensial 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 kredensial 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 [GetSessionToken](#) di AWS SDK for Python (Boto3) Referensi API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan IAM dengan SDK AWS](#). Topik ini juga mencakup informasi tentang cara memulai dan detail versi-versi SDK sebelumnya.

Keamanan di IAM dan AWS STS

Keamanan cloud di AWS merupakan prioritas tertinggi. Sebagai pelanggan AWS, Anda mendapatkan manfaat dari pusat data dan arsitektur jaringan yang dibangun untuk memenuhi persyaratan organisasi yang paling sensitif terhadap keamanan.

Keamanan adalah tanggung jawab bersama antara AWS dan Anda. [Model tanggung jawab bersama](#) menggambarkan ini sebagai keamanan dari cloud dan keamanan di dalam cloud:

- Keamanan cloud – AWS bertanggung jawab untuk melindungi infrastruktur yang menjalankan layanan-layanan AWS di dalam AWS Cloud. AWS juga memberikan Anda layanan yang dapat digunakan dengan aman. Auditor pihak ketiga melakukan pengujian dan verifikasi secara berkala terhadap efektivitas keamanan kami sebagai bagian dari [Program Kepatuhan AWS](#). Untuk mempelajari tentang program kepatuhan yang berlaku di AWS Identity and Access Management (IAM), lihat [AWS Layanan dalam Cakupan melalui Program Kepatuhan AWS](#).
- Keamanan di cloud – Tanggung jawab Anda ditentukan menurut layanan AWS 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 mengonfigurasi IAM dan AWS STS untuk memenuhi tujuan keamanan dan kepatuhan Anda. Anda juga mempelajari cara menggunakan AWS layanan lain yang membantu Anda memantau dan mengamankan sumber daya IAM Anda.

Konten

- [Kredensial keamanan AWS](#)
- [AWS pedoman audit keamanan](#)
- [Perlindungan data di AWS Identity and Access Management](#)
- [Pencatatan 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 dalam AWS Identity and Access Management](#)
- [Analisis konfigurasi dan kerentanan dalam AWS Identity and Access Management](#)
- [AWS kebijakan terkelola untuk AWS Identity and Access Management Access Analyzer](#)

Kredensial keamanan AWS

Saat Anda berinteraksi dengan AWS, Anda menentukan kredensial keamanan AWS untuk memverifikasi siapa Anda dan apakah memiliki izin untuk mengakses sumber daya yang Anda minta. AWS menggunakan kredensial 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. Semua pengguna AWS memiliki kredensial keamanan. Ada pemilik akun (pengguna root), pengguna federasi pengguna di Pusat Identitas AWS IAM, dan pengguna IAM.

Pengguna memiliki kredensi keamanan jangka panjang atau sementara. Pengguna root, pengguna IAM, dan kunci akses memiliki kredensial keamanan jangka panjang yang tidak kedaluwarsa. Untuk melindungi kredensi jangka panjang, ada proses untuk [mengelola kunci akses](#), [mengubah kata sandi](#), dan mengaktifkan [MFA](#).

Peran IAM, pengguna di Pusat Identitas AWS IAM, dan pengguna federasi memiliki kredensial keamanan sementara. Kredensial keamanan sementara kedaluwarsa setelah jangka waktu tertentu atau ketika pengguna mengakhiri sesi mereka. Kredensial sementara bekerja hampir identik dengan kredensial 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 mengenali mereka atau mengizinkan akses apa pun dari permintaan API yang dibuat bersama mereka.
- 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 sebagai berikut dibandingkan kredensial jangka panjang:

- Anda tidak perlu mendistribusikan atau menyimpan kredensial keamanan AWS jangka panjang dengan aplikasi.
- Anda dapat memberikan akses ke sumber daya AWS Anda kepada pengguna tanpa harus menentukan identitas AWS 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 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 AndaAkun AWS:

- Saat Anda membuat Akun AWS, kami membuat pengguna root akun. Kredensial 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.
- Anda tidak dapat menggunakan kebijakan IAM untuk menolak akses pengguna root ke sumber daya secara eksplisit. Anda hanya dapat menggunakan [kebijakan kontrol layanan \(SCP\)](#) AWS Organizations 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, lihat [Tugas yang memerlukan kredensial pengguna root](#).
- Kredensial keamanan adalah khusus akun. Jika Anda memiliki akses ke beberapa Akun AWS, Anda memiliki kredensial 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

AndaAkun AWS. Jika Anda harus mengubah atau mencabut izin dalam menanggapi 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 kredensi sementara yang disediakan oleh peran IAM dan pengguna federasi, bukan kredensial jangka panjang yang disediakan oleh pengguna IAM dan kunci akses.

Identitas federasi

Identitas federasi adalah pengguna dengan identitas eksternal yang diberikan AWS kredensial sementara yang dapat mereka gunakan untuk mengakses sumber daya yang aman. Akun AWS Identitas eksternal dapat berasal dari toko identitas perusahaan (seperti LDAP atau Windows Active Directory) atau dari pihak ketiga (seperti Login dengan Amazon, Facebook, atau Google). Identitas federasi tidak masuk dengan portal AWS Management Console atau AWS akses.

Untuk mengaktifkan identitas federasi untuk masuk ke AWS, Anda harus membuat URL khusus yang menyertakan `https://signin.aws.amazon.com/federation` Untuk informasi selengkapnya, lihat [Mengaktifkan akses broker identitas khusus ke konsol AWS](#).

Untuk informasi lebih lanjut tentang identitas federasi, lihat [Penyedia dan federasi identitas](#)

Autentikasi multi-faktor (MFA)

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 kredensial dan semua pengguna IAM. Untuk informasi selengkapnya, lihat [Menggunakan otentikasi multi-faktor \(MFA\) di AWS](#).

Saat mengaktifkan MFA dan masuk ke MFAAkun AWS, Anda akan diminta untuk mendapatkan kredensial masuk, ditambah respons yang dihasilkan oleh perangkat MFA, seperti kode, sentuhan atau ketukan, atau pemindaian biometrik. Saat Anda menambahkan MFA, Akun AWS pengaturan dan sumber daya Anda lebih aman.

Secara default, MFA tidak diaktifkan. Anda dapat mengaktifkan dan mengelola perangkat MFA untuk Pengguna root akun AWS dengan membuka halaman [kredensial Keamanan atau dasbor IAM](#) di AWS

Management Console Untuk informasi selengkapnya tentang mengaktifkan MFA untuk pengguna IAM, lihat. [Mengaktifkan perangkat MFA untuk pengguna di AWS](#)

Untuk informasi selengkapnya tentang masuk dengan perangkat autentikasi multi-faktor (MFA), lihat. [Menggunakan perangkat MFA dengan halaman masuk IAM Anda](#)

Akses terprogram

Anda memberikan kunci AWS akses Anda untuk melakukan panggilan terprogram ke AWS atau untuk menggunakan AWS Command Line Interface atau AWS Tools for PowerShell. Sebaiknya gunakan kunci akses jangka pendek jika memungkinkan.

Ketika 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 kredensialnya kedaluwarsa. Setelah kedaluwarsa, mereka tidak lagi valid.

ID kunci akses yang 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 kredensial sementara yang Anda buat menggunakan AWS STS operasi.

Pengguna membutuhkan akses terprogram jika mereka ingin berinteraksi dengan AWS luar. AWS Management Console Cara untuk memberikan akses terprogram tergantung pada jenis pengguna yang mengakses AWS.

Untuk memberikan akses terprogram kepada pengguna, pilih salah satu opsi berikut.

Pengguna mana yang membutuhkan akses terprogram?	Ke	oleh
Identitas tenaga kerja (Pengguna dikelola di Pusat Identitas IAM)	Gunakan kredensial sementara untuk menandatangani permintaan terprogram ke AWS CLI, AWS SDK, atau API. AWS	Mengikuti instruksi untuk antarmuka yang ingin Anda gunakan. <ul style="list-style-type: none"> • Untuk AWS CLI, lihat Mengonfigurasi yang akan AWS CLI digunakan AWS IAM Identity Center dalam Panduan AWS Command Line Interface Pengguna. • Untuk AWS SDK, alat, dan AWS API, lihat otentikasi Pusat Identitas IAM di Panduan Referensi AWS SDK dan Alat.
IAM	Gunakan kredensial sementara untuk menandatangani permintaan terprogram ke AWS CLI, AWS SDK, atau API. AWS	Mengikuti petunjuk dalam Menggunakan kredensial sementara dengan AWS sumber daya di Panduan Pengguna IAM.
IAM	(Tidak disarankan) Gunakan kredensial jangka panjang untuk menandatangani permintaan terprogram ke AWS CLI, AWS SDK, atau API. AWS	Mengikuti instruksi untuk antarmuka yang ingin Anda gunakan. <ul style="list-style-type: none"> • Untuk mengetahui AWS CLI, lihat Mengotentikasi menggunakan kredensial pengguna IAM di Panduan Pengguna. AWS Command Line Interface

Pengguna mana yang membutuhkan akses terprogram?	Ke	oleh
		<ul style="list-style-type: none">• Untuk AWS SDK dan alat bantu, lihat Mengautentikasi menggunakan kredensi jangka panjang di Panduan Referensi AWS SDK dan Alat.• Untuk AWS API, lihat Mengelola 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 menanamkan kunci akses jangka panjang dan kunci akses rahasia dalam kode aplikasi Anda atau dalam 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 AWS Secrets Manager?](#) dalam AWS Secrets Manager Panduan Pengguna.
- Gunakan peran IAM untuk menghasilkan kredensial keamanan sementara bila memungkinkan — Selalu 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 yang 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 Peran Di Mana Saja](#).

- Gunakan alternatif untuk kunci akses jangka panjang untuk AWS Command Line Interface (AWS CLI) atau **aws-shell** — Alternatif termasuk yang berikut ini.
 - AWS CloudShell adalah shell pra-otentikasi berbasis browser yang dapat Anda luncurkan langsung dari file. AWS Management Console 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 lebih lanjut, lihat [Apa AWS CloudShell?](#) dalam AWS CloudShell Panduan Pengguna.
 - AWS CLI Integrasi versi 2 dengan AWS IAM Identity Center (IAM Identity Center). Anda dapat mengotentikasi pengguna dan memberikan kredensi jangka pendek untuk menjalankan perintah. AWS CLI Untuk mempelajari lebih lanjut, lihat [Mengintegrasikan AWS CLI dengan Pusat Identitas IAM](#) di Panduan AWS IAM Identity Center Pengguna dan [Mengonfigurasi Pusat Identitas IAM AWS CLI untuk menggunakan](#) di Panduan Pengguna. AWS Command Line Interface
- Jangan membuat kunci akses jangka panjang untuk pengguna manusia yang memerlukan akses ke aplikasi atau Layanan AWS — IAM Identity Center dapat menghasilkan kredensial akses sementara untuk diakses oleh pengguna iDP eksternal Anda. Layanan AWS Ini menghilangkan kebutuhan untuk membuat dan mengelola kredensi jangka panjang di IAM. Di Pusat Identitas IAM, buat set izin Pusat Identitas IAM yang memberikan akses pengguna iDP eksternal. Kemudian tetapkan grup dari IAM Identity Center ke set izin yang dipilih. Akun AWS Untuk informasi selengkapnya, lihat [Apa itu AWS IAM Identity Center](#), [Connect ke penyedia identitas eksternal Anda](#), dan [set 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 membuat profil instans yang dilampirkan ke instans Amazon EC2, Anda dapat menetapkan AWS peran ke instans dan membuatnya tersedia untuk semua aplikasinya. Profil instans berisi peran dan memungkinkan program yang berjalan di instans Amazon EC2 untuk mendapatkan kredensial sementara. Untuk mempelajari lebih lanjut, lihat [Menggunakan peran IAM untuk memberikan izin ke aplikasi yang berjalan di instans Amazon EC2](#).

Mengakses AWS menggunakan kredensi Anda AWS

AWS memerlukan berbagai jenis kredensi keamanan, tergantung pada bagaimana Anda mengakses AWS dan jenis AWS pengguna Anda. Misalnya, Anda menggunakan kredensial masuk untuk AWS Management Console sementara Anda menggunakan kunci akses untuk melakukan panggilan

terprogram. AWS Selain itu, setiap identitas yang Anda gunakan, apakah itu pengguna root akun, pengguna AWS Identity and Access Management (IAM), pengguna, atau identitas federasi, memiliki kredensi unik di dalamnya. AWS IAM Identity Center AWS

Untuk step-by-step petunjuk tentang cara masuk AWS sesuai dengan jenis pengguna, lihat [Cara masuk AWS di](#) Panduan Pengguna AWS Masuk.

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 pedoman 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](#) Hub Keamanan menggunakan kontrol keamanan untuk mengevaluasi konfigurasi sumber daya dan standar keamanan guna membantu Anda mematuhi berbagai kerangka kerja kepatuhan. Untuk informasi selengkapnya tentang penggunaan Security Hub untuk mengevaluasi sumber daya IAM, lihat [kontrol AWS Identity and Access Management](#) di Panduan AWS Security Hub Pengguna.

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, AWS OpsWorks tumpukan, templat, dll. AWS 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 kredensial 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 pengguna 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 API yang 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, konfirmasi 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 bersama](#) AWS diterapkan untuk perlindungan data AWS Identity and Access Management. Sebagaimana dijelaskan dalam model ini, AWS bertanggung jawab untuk melindungi infrastruktur global yang menjalankan semua AWS Cloud. Anda bertanggung jawab untuk memelihara kendali atas isi yang dihost pada infrastruktur ini. Anda juga bertanggung jawab atas tugas-tugas konfigurasi dan manajemen keamanan untuk Layanan AWS yang Anda gunakan. Lihat informasi yang lebih lengkap tentang privasi data dalam [Pertanyaan Umum Privasi Data](#). Lihat informasi tentang perlindungan data di Eropa di pos blog [Model Tanggung Jawab Bersama dan GDPR AWS](#) di Blog Keamanan AWS.

Untuk tujuan perlindungan data, sebaiknya lindungi kredensial Akun AWS dan siapkan untuk masing-masing pengguna 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 sumber daya AWS. Kami mensyaratkan TLS 1.2 dan menganjurkan TLS 1.3.
- Siapkan API dan pengelolan aktivitas pengguna dengan AWS CloudTrail.
- Gunakan solusi enkripsi AWS, bersama semua kontrol keamanan bawaan dalam Layanan AWS.
- Gunakan layanan keamanan terkelola lanjut seperti Amazon Macie, yang membantu menemukan dan mengamankan data sensitif yang disimpan di Amazon S3.
- Jika Anda memerlukan modul kriptografi tervalidasi FIPS 140-2 ketika mengakses AWS melalui antarmuka baris perintah atau API, gunakan titik akhir FIPS. Lihat informasi yang lebih lengkap tentang titik akhir FIPS yang tersedia di [Standar Pemrosesan Informasi Federal \(FIPS\) 140-2](#).

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, APIAWS 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 pengenalan 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 dalam bergerak

Data identifikasi pelanggan, termasuk kata sandi, dienkripsi dalam perjalanan menggunakan TLS 1.2 dan 1.3. Semua titik akhir AWS STS mendukung HTTPS untuk mengenkripsi data saat transit. Untuk daftar titik akhir AWS STS, lihat [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 layanan AWS STS menggunakan VPC endpoint. Untuk mempelajari informasi lebih lanjut, lihat [Titik akhir VPC antarmuka](#).

Pencatatan dan pemantauan di AWS Identity and Access Management

Pemantauan adalah bagian penting dari pemeliharaan keandalan, ketersediaan, dan performa AWS Identity and Access Management (IAM), AWS Security Token Service (AWS STS) dan AWS solusi Anda lainnya. AWS menyediakan beberapa alat untuk memantau AWS sumber daya Anda dan merespons potensi insiden:

- AWS CloudTrail menangkap semua panggilan API untuk IAM dan AWS STS sebagai peristiwa, termasuk panggilan dari konsol dan dari panggilan API. Untuk mempelajari lebih lanjut tentang menggunakan CloudTrail dengan IAM dan AWS STS, lihat [Mencatat panggilan IAM dan AWS STS 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 IAM role, yang dibagi 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 di secara langsung. 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 instans Amazon EC2 Anda dan secara otomatis meluncurkan instans baru ketika diperlukan. Untuk informasi selengkapnya, lihat [Panduan CloudWatch Pengguna Amazon](#).
- Amazon CloudWatch Logs membantu Anda memantau, menyimpan, dan mengakses file log dari instans Amazon EC2 CloudTrail, dan sumber lainnya. CloudWatch Log dapat memantau informasi dalam file log dan memberi tahu Anda ketika ambang tertentu terpenuhi. Anda juga dapat mengarsipkan data log Anda dalam penyimpanan yang sangat tahan lama. 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 AWS Identity and Access Management](#).

Validasi kepatuhan untuk AWS Identity and Access Management

Auditor pihak ketiga 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.

Untuk mempelajari apakah an Layanan AWS berada dalam lingkup program kepatuhan tertentu, lihat [Layanan AWS di Lingkup oleh Program Kepatuhan Layanan AWS](#) dan pilih program kepatuhan yang Anda minati. Untuk informasi umum, lihat [Program Kepatuhan AWS](#).

Anda bisa 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, serta hukum dan peraturan yang berlaku. AWS menyediakan sumber daya berikut untuk membantu kepatuhan:

- [Panduan Memulai Cepat Keamanan dan Kepatuhan — Panduan](#) penerapan ini membahas pertimbangan arsitektur dan memberikan langkah-langkah untuk menerapkan lingkungan dasar AWS yang berfokus pada keamanan dan kepatuhan.
- [Arsitektur untuk Keamanan dan Kepatuhan HIPAA di Amazon Web Services](#) — Whitepaper ini menjelaskan bagaimana perusahaan dapat menggunakan AWS untuk membuat aplikasi yang memenuhi syarat HIPAA.

Note

Tidak semua Layanan AWS memenuhi syarat HIPAA. Untuk informasi selengkapnya, lihat [Referensi Layanan yang Memenuhi Syarat HIPAA](#).

- [Sumber Daya Kepatuhan AWS](#) – Kumpulan buku kerja dan panduan ini mungkin berlaku untuk industri dan lokasi Anda.
- [AWS Panduan Kepatuhan Pelanggan](#) - Memahami model tanggung jawab bersama melalui lensa kepatuhan. Panduan ini merangkum praktik terbaik untuk mengamankan Layanan AWS dan memetakan panduan untuk kontrol keamanan di berbagai kerangka kerja (termasuk Institut Standar dan Teknologi Nasional (NIST), Dewan Standar Keamanan Industri Kartu Pembayaran (PCI), dan Organisasi Internasional untuk Standardisasi (ISO)).

- [Mengevaluasi Sumber Daya dengan Aturan](#) di Panduan Developer AWS Config – Layanan AWS Config menilai seberapa baik konfigurasi sumber daya Anda dalam mematuhi praktik-praktik internal, pedoman industri, dan regulasi internal.
- [AWS Security Hub](#)— Ini Layanan AWS memberikan pandangan komprehensif tentang keadaan keamanan Anda di dalamnya AWS. Security Hub menggunakan kontrol keamanan untuk mengevaluasi AWS sumber daya Anda dan untuk memeriksa kepatuhan Anda terhadap standar industri keamanan dan praktik terbaik. Untuk daftar layanan dan kontrol yang didukung, lihat [referensi kontrol Security Hub](#).
- [AWS Audit Manager](#) – Layanan AWS ini akan membantu Anda untuk terus-menerus mengaudit penggunaan AWS untuk menyederhanakan bagaimana Anda mengelola risiko dan kepatuhan terhadap regulasi dan standar industri.

Ketahanan di AWS Identity and Access Management

Infrastruktur global AWS dibangun di sekitar Wilayah AWS dan Availability Zone. 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 Wilayah AWS dan Zona Ketersediaan, lihat [Infrastruktur Global AWS](#).

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.

AWS STS permintaan selalu pergi ke satu titik akhir global secara default. Anda dapat menggunakan AWS STS endpoint Regional untuk mengurangi latensi atau memberikan redundansi tambahan untuk aplikasi Anda. Untuk mempelajari selengkapnya, lihat [Mengelola AWS STS dalam 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 kredensial 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 API, 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 [kredensial sementara, bukan kunci](#) akses jangka panjang.

Kredensial sementara

Anda dapat [meminta kredensial sementara baru](#) dengan [titik akhir layanan AWS STS](#) Regional setidaknya selama 24 jam. Operasi API berikut menghasilkan kredensial 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](#).

AWS Management Console

- Anda mungkin dapat menggunakan titik akhir masuk Regional untuk masuk ke pengguna IAM AWS Management Console 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?region=us-west-2`

- Anda mungkin tidak dapat menyelesaikan otentikasi multi-faktor [Universal 2nd Factor \(U2F\)](#) (MFA).

Praktik terbaik untuk ketahanan IAM

AWStelah 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 dalam AWS Identity and Access Management

Sebagai layanan terkelola, Manajemen AWS Identitas dan Akses dilindungi oleh keamanan jaringan AWS global. Untuk informasi tentang layanan AWS keamanan dan cara AWS melindungi infrastruktur, lihat [Keamanan AWS Cloud](#). Untuk merancang AWS lingkungan Anda menggunakan praktik terbaik untuk keamanan infrastruktur, lihat [Perlindungan](#) Infrastruktur dalam Kerangka Kerja Pilar Keamanan yang AWS Diarsiteksikan dengan Baik.

Anda menggunakan panggilan API yang AWS dipublikasikan untuk mengakses IAM melalui jaringan. Klien harus mendukung hal berikut:

- Transport Layer Security (TLS). Kami membutuhkan TLS 1.2 dan merekomendasikan TLS 1.3.
- Suite cipher dengan kerahasiaan maju sempurna (PFS) seperti DHE (Ephemeral Diffie-Hellman) atau ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Sebagian besar sistem modern seperti Java 7 dan sistem yang lebih baru mendukung mode ini.

Selain itu, permintaan harus ditandatangani menggunakan access key ID dan secret access key yang terkait dengan principal IAM. Atau Anda bisa menggunakan [AWS Security Token Service](#) (AWS STS) untuk membuat kredensial keamanan sementara guna menandatangani permintaan.

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 VPC tertentu. Secara efektif, ini mengisolasi akses jaringan ke sumber daya IAM yang diberikan hanya dari VPC tertentu dalam jaringan. AWS

Analisis konfigurasi dan kerentanan dalam AWS Identity and Access Management

AWS menangani tugas-tugas keamanan basic seperti sistem operasi tamu (OS) dan patching basis data, 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: Ikhtisar Proses Keamanan](#) (kertas putih)

Sumber daya berikut juga membahas konfigurasi dan analisis kerentanan diAWS Identity and Access Management(IAM):

- [Validasi kepatuhan untuk AWS Identity and Access Management](#)
- [Praktik terbaik keamanan dan kasus penggunaan AWS Identity and Access Management](#)

AWS kebijakan terkelola untuk AWS Identity and Access Management Access Analyzer

Kebijakan AWS terkelola adalah kebijakan mandiri yang dibuat dan dikelola oleh AWS. AWS Kebijakan terkelola dirancang untuk memberikan izin bagi banyak kasus penggunaan umum sehingga Anda dapat mulai menetapkan izin kepada pengguna, grup, dan peran.

Perlu diingat bahwa kebijakan AWS terkelola mungkin tidak memberikan izin hak istimewa paling sedikit untuk kasus penggunaan spesifik Anda karena tersedia untuk digunakan semua pelanggan. AWS 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 AWS terkelola saat baru Layanan AWS diluncurkan atau operasi API baru tersedia untuk layanan yang ada.

Untuk informasi selengkapnya, lihat [Kebijakan yang dikelola AWS](#) dalam Panduan Pengguna IAM.

IAM ReadOnlyAccess

Gunakan kebijakan IAMReadOnlyAccess terkelola 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 IAM Access Analyzer. Lihat [kebijakan](#) untuk daftar lengkap layanan dan tindakan yang didukung oleh kebijakan ini.

IAM UserChangePassword

Gunakan kebijakan `IAMUserChangePassword` terkelola untuk memungkinkan pengguna IAM mengubah kata sandi mereka.

Anda mengonfigurasi pengaturan Akun IAM dan kebijakan Kata Sandi untuk memungkinkan pengguna IAM mengubah kata sandi akun IAM mereka. Saat Anda mengizinkan tindakan ini, IAM melampirkan kebijakan berikut ke setiap pengguna:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iam:ChangePassword"
      ],
      "Resource": [
        "arn:aws:iam::*:user/${aws:username}"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "iam:GetAccountPasswordPolicy"
      ],
      "Resource": "*"
    }
  ]
}
```

IAM AccessAnalyzerFullAccess

Gunakan kebijakan `IAMAccessAnalyzerFullAccess` AWS terkelola untuk mengizinkan administrator mengakses IAM Access Analyzer.

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 membuat [peran terkait layanan](#), yang memungkinkan IAM Access Analyzer 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 IAM Access Analyzer 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.

```
{
  "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": "*"
}
]
```

IAM AccessAnalyzerReadOnlyAccess

Gunakan kebijakan `IAMAccessAnalyzerReadOnlyAccess` AWS terkelola untuk mengizinkan akses hanya-baca ke IAM Access Analyzer.

Untuk juga mengizinkan akses hanya-baca ke IAM Access Analyzer AWS Organizations, buat kebijakan terkelola pelanggan yang memungkinkan tindakan Deskripsikan dan Daftar dari kebijakan terkelola. [IAM AccessAnalyzerFullAccess](#) AWS

Izin tingkat layanan

Kebijakan ini menyediakan akses hanya-baca ke IAM Access Analyzer. Tidak ada izin layanan lain yang disertakan dalam kebijakan ini.

```
{
  "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 pada peran terkait layanan yang memungkinkan IAM Access Analyzer untuk melakukan tindakan atas nama Anda. Untuk informasi selengkapnya, lihat [Menggunakan peran terkait layanan untuk AWS Identity and Access Management Access Analyzer](#).

Pengelompokan izin

Kebijakan ini memungkinkan akses ke IAM Access Analyzer untuk menganalisis metadata sumber daya dari beberapa. Layanan AWS

- Amazon DynamoDB - Memungkinkan izin untuk melihat aliran dan tabel DynamoDB.
- Amazon Elastic Compute Cloud - Memungkinkan izin untuk mendeskripsikan alamat IP, snapshot, dan VPC.
- Amazon Elastic Container Registry - Memungkinkan izin untuk mendeskripsikan repositori gambar dan mengambil kebijakan 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 pengguna, profil login, kunci akses, dan layanan data yang terakhir diakses.
- AWS Key Management Service— Memungkinkan izin untuk melihat informasi rinci tentang kunci KMS dan kebijakan dan hibah utamanya.
- AWS Lambda— Memungkinkan izin untuk melihat informasi tentang alias, fungsi, lapisan, dan alias Lambda.
- AWS Organizations— Memungkinkan izin untuk Organizations dan memungkinkan pembuatan analyzer dalam AWS organisasi sebagai zona kepercayaan.
- Amazon Relational Database Service - 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 jalur akses Amazon S3, bucket, dan bucket direktori Amazon S3 yang menggunakan kelas penyimpanan Amazon S3 Express One.
- AWS Secrets Manager— Memungkinkan izin untuk melihat informasi rinci tentang rahasia dan kebijakan sumber daya yang melekat pada rahasia.

- Amazon Simple Notification Service - Memungkinkan izin untuk melihat informasi rinci tentang suatu topik.
- Amazon Simple Queue Service - Memungkinkan izin untuk melihat informasi rinci tentang antrian tertentu.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AccessAnalyzerServiceRolePolicy",
      "Effect": "Allow",
      "Action": [
        "dynamodb:GetResourcePolicy",
        "dynamodb:ListStreams",
        "dynamodb:ListTables",
        "ec2:DescribeAddresses",
        "ec2:DescribeByoipCidrs",
        "ec2:DescribeSnapshotAttribute",
        "ec2:DescribeSnapshots",
        "ec2:DescribeVpcEndpoints",
        "ec2:DescribeVpcs",
        "ec2:GetSnapshotBlockPublicAccessState",
        "ecr:DescribeRepositories",
        "ecr:GetRepositoryPolicy",
        "elasticfilesystem:DescribeFileSystemPolicy",
        "elasticfilesystem:DescribeFileSystems",
        "iam:GenerateServiceLastAccessedDetails",
        "iam:GetAccessKeyLastUsed",
        "iam:GetGroup",
        "iam:GetLoginProfile",
        "iam:GetRole",
        "iam:GetServiceLastAccessedDetails",
        "iam:GetUser",
        "iam:ListAccessKeys",
        "iam:ListEntitiesForPolicy",
        "iam:ListRoles",
        "iam:ListUsers",
        "kms:DescribeKey",
        "kms:GetKeyPolicy",
        "kms:ListGrants",
        "kms:ListKeyPolicies",
        "kms:ListKeys",
```

```
"lambda:GetFunctionUrlConfig",
"lambda:GetLayerVersionPolicy",
"lambda:GetPolicy",
"lambda:ListAliases",
"lambda:ListFunctions",
"lambda:ListLayers",
"lambda:ListLayerVersions",
"lambda:ListVersionsByFunction",
"organizations:DescribeAccount",
"organizations:DescribeOrganization",
"organizations:DescribeOrganizationalUnit",
"organizations:ListAccounts",
"organizations:ListAccountsForParent",
"organizations:ListAWSServiceAccessForOrganization",
"organizations:ListChildren",
"organizations:ListDelegatedAdministrators",
"organizations:ListOrganizationalUnitsForParent",
"organizations:ListParents",
"organizations:ListRoots",
"rds:DescribeDBClusterSnapshotAttributes",
"rds:DescribeDBClusterSnapshots",
"rds:DescribeDBSnapshotAttributes",
"rds:DescribeDBSnapshots",
"s3:DescribeMultiRegionAccessPointOperation",
"s3:GetAccessPoint",
"s3:GetAccessPointPolicy",
"s3:GetAccessPointPolicyStatus",
"s3:GetAccountPublicAccessBlock",
"s3:GetBucketAcl",
"s3:GetBucketLocation",
"s3:GetBucketPolicyStatus",
"s3:GetBucketPolicy",
"s3:GetBucketPublicAccessBlock",
"s3:GetMultiRegionAccessPoint",
"s3:GetMultiRegionAccessPointPolicy",
"s3:GetMultiRegionAccessPointPolicyStatus",
"s3:ListAccessPoints",
"s3:ListAllMyBuckets",
"s3:ListMultiRegionAccessPoints",
"s3express:GetBucketPolicy",
"s3express:ListAllMyDirectoryBuckets",
"sns:GetTopicAttributes",
"sns:ListTopics",
"secretsmanager:DescribeSecret",
```

```

    "secretsmanager:GetResourcePolicy",
    "secretsmanager:ListSecrets",
    "sqs:GetQueueAttributes",
    "sqs:ListQueues"
  ],
  "Resource": "*"
}
]
}

```

IAM dan IAM Access Analyzer memperbarui kebijakan terkelola AWS

Lihat detail tentang pembaruan IAM dan kebijakan AWS terkelola 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.

Perubahan	Deskripsi	Tanggal
AccessAnalyzerServiceRolePolicy — Izin Ditambahkan	IAM Access Analyzer menambahkan dukungan untuk izin untuk mengambil status saat ini dari blokir akses publik untuk snapshot Amazon EC2 ke izin tingkat layanan. AccessAnalyzerServiceRolePolicy	23 Januari 2024
AccessAnalyzerServiceRolePolicy — Izin Ditambahkan	IAM Access Analyzer menambahkan dukungan untuk aliran dan tabel DynamoDB ke izin tingkat layanan. AccessAnalyzerServiceRolePolicy	Januari 11, 2024

Perubahan	Deskripsi	Tanggal
AccessAnalyzerServiceRolePolicy — Izin Ditambahkan	IAM Access Analyzer menambahkan dukungan untuk bucket direktori Amazon S3 ke izin tingkat layanan. <code>AccessAnalyzerServiceRolePolicy</code>	1 Desember 2023
IAM AccessAnalyzerReadOnlyAccess - Izin Ditambahkan	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 — Izin Ditambahkan	IAM Access Analyzer menambahkan tindakan IAM ke izin tingkat layanan untuk mendukung tindakan berikut: <code>AccessAnalyzerServiceRolePolicy</code> <ul style="list-style-type: none"> • Daftar entitas untuk kebijakan • Menghasilkan detail layanan yang terakhir diakses • Daftar informasi kunci akses 	26 November 2023

Perubahan	Deskripsi	Tanggal
AccessAnalyzerServiceRolePolicy — Izin Ditambahkan	<p>IAM Access Analyzer menambahkan dukungan untuk jenis sumber daya berikut ke izin tingkat layanan: <code>AccessAnalyzerServiceRolePolicy</code></p> <ul style="list-style-type: none"> • Cuplikan volume Amazon EBS • Repositori Amazon ECR • Sistem file Amazon EFS • Cuplikan Amazon RDS DB • Cuplikan cluster Amazon RDS DB • Topik Amazon SNS 	25 Oktober 2022
AccessAnalyzerServiceRolePolicy — Izin Ditambahkan	<p>IAM Access Analyzer menambahkan <code>lambda:GetFunctionUrlConfig</code> tindakan ke izin tingkat layanan. <code>AccessAnalyzerServiceRolePolicy</code></p>	April 6, 2022
AccessAnalyzerServiceRolePolicy — Izin Ditambahkan	<p>IAM Access Analyzer menambahkan tindakan Amazon S3 baru untuk menganalisis metadata yang terkait dengan titik akses multi-wilayah.</p>	2 September 2021

Perubahan	Deskripsi	Tanggal
IAM AccessAnalyzerRead OnlyAccess - Izin Ditambahkan	<p>IAM Access Analyzer menambahkan tindakan baru untuk memberikan <code>ValidatePolicy</code> izin agar Anda dapat menggunakan pemeriksaan kebijakan untuk validasi.</p> <p>Izin ini diperlukan oleh IAM Access Analyzer untuk melakukan pemeriksaan kebijakan pada kebijakan Anda.</p>	16 Maret 2021
IAM Access Analyzer mulai melacak perubahan	IAM Access Analyzer mulai melacak perubahan untuk kebijakan yang AWS dikelola.	1 Maret 2021

Menggunakan 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.
- IAM Access Analyzer penganalisis akses 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 praktik terbaik. AWS
- Pemeriksaan kebijakan khusus 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 jika akses tidak diinginkan dan risiko keamanan. Selain membantu mengidentifikasi sumber daya yang dibagikan dengan entitas eksternal, Anda dapat menggunakan temuan IAM Access Analyzer untuk melihat pratinjau bagaimana kebijakan Anda memengaruhi akses publik dan lintas akun ke sumber daya Anda sebelum menerapkan izin sumber daya. Temuan ini 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, lihat [Melihat dasbor temuan IAM Access Analyzer](#).

Note

Entitas eksternal dapat berupa AWS akun lain, pengguna root, pengguna atau peran IAM, pengguna federasi, AWS layanan, pengguna anonim, atau entitas lain yang dapat Anda gunakan untuk membuat filter. Untuk informasi lebih lanjut, lihat [Elemen Kebijakan JSON AWS : Penanggung Jawab](#)

Saat mengaktifkan IAM Access Analyzer, Anda membuat analisa 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 akan 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 rincian tentang sumber daya, entitas eksternal dengan akses ke sana, dan izin yang diberikan sehingga Anda dapat mengambil tindakan yang tepat. Anda dapat melihat detail yang disertakan dalam temuan untuk menentukan apakah akses sumber daya disengaja atau merupakan potensi risiko yang harus Anda selesaikan. Saat menambahkan kebijakan ke sumber daya, atau memperbarui kebijakan yang ada, IAM Access Analyzer akan 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 keterlambatan dalam temuan yang dihasilkan. IAM Access Analyzer 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 jalur akses multi-wilayah. Selain itu, jika ada masalah pengiriman dengan pengiriman AWS CloudTrail log, perubahan kebijakan tidak memicu pemindaian ulang sumber daya yang

dilaporkan dalam temuan tersebut. Ketika ini terjadi, IAM Access Analyzer menganalisis kebijakan baru atau yang diperbarui selama pemindaian berkala berikutnya, yaitu dalam 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 menggunakan tautan Pindai Ulang di halaman detail Temuan, atau dengan menggunakan [StartResourceScan](#) pengoperasian API IAM Access Analyzer. Untuk mempelajari selengkapnya, lihat [Menyelesaikan temuan](#).

⚠ Important

IAM Access Analyzer hanya menganalisis kebijakan yang diterapkan pada sumber daya di AWS Wilayah yang sama dengan yang diaktifkan. Untuk memantau semua sumber daya di AWS lingkungan Anda, Anda harus membuat penganalisis untuk mengaktifkan IAM Access Analyzer di setiap Wilayah tempat Anda menggunakan sumber daya yang didukung. AWS

IAM Access Analyzer menganalisis jenis sumber daya berikut:

- [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 yang tidak terpakai 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 ini memberikan visibilitas ke layanan dan tindakan yang tidak digunakan.

Temuan untuk akses eksternal dan penganalisis akses yang tidak digunakan diatur ke dalam dasbor ringkasan visual. Dasbor menyoroti Anda Akun AWS yang memiliki temuan paling banyak dan memberikan rincian temuan berdasarkan jenisnya. Untuk informasi selengkapnya tentang dasbor, lihat [Melihat dasbor temuan IAM Access Analyzer](#).

IAM Access Analyzer meninjau informasi yang terakhir diakses untuk semua peran di AWS organisasi dan akun Anda untuk membantu Anda mengidentifikasi akses yang tidak digunakan. Informasi terakhir yang diakses tindakan IAM membantu Anda mengidentifikasi tindakan yang tidak digunakan untuk peran dalam Anda. Akun AWS Untuk informasi selengkapnya, lihat [Menyaring izin di AWS menggunakan informasi yang terakhir diakses](#).

Memvalidasi kebijakan terhadap praktik AWS terbaik

Anda dapat memvalidasi kebijakan Anda 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 editor kebijakan AWS CLI, AWS API, atau 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 terbaik. AWS Untuk mempelajari lebih lanjut tentang memvalidasi kebijakan menggunakan validasi kebijakan, lihat [Validasi kebijakan IAM Access Analyzer](#)

Memvalidasi kebijakan terhadap standar keamanan yang Anda tentukan

Anda dapat memvalidasi kebijakan Anda terhadap standar keamanan yang ditentukan menggunakan pemeriksaan kebijakan khusus IAM Access Analyzer. Anda dapat membuat atau mengedit kebijakan menggunakan editor kebijakan AWS CLI, AWS API, atau JSON di konsol IAM. Melalui konsol, Anda dapat memeriksa apakah kebijakan yang diperbarui memberikan akses baru dibandingkan dengan versi yang ada. Melalui AWS CLI dan AWS API, Anda juga dapat memeriksa tindakan IAM tertentu yang Anda anggap 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 sesuai dengan standar keamanan Anda. Untuk mempelajari selengkapnya tentang memvalidasi kebijakan menggunakan pemeriksaan kebijakan khusus, lihat [Pemeriksaan kebijakan khusus IAM Access Analyzer](#).

Membuat 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 ditentukan. Kemudian menghasilkan kebijakan IAM yang didasarkan pada aktivitas akses tersebut. Anda dapat menggunakan kebijakan yang dihasilkan untuk menyaring izin entitas dengan melampirkannya ke pengguna atau peran IAM. Untuk mempelajari selengkapnya tentang membuat kebijakan menggunakan IAM Access Analyzer, lihat [Pembuatan kebijakan IAM Access Analyzer](#)

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 terpakai yang Anda buat.
- Membuat penganalisis akses yang tidak digunakan di beberapa Wilayah akan mengakibatkan Anda dikenakan biaya untuk setiap penganalisis.
- Peran terkait layanan tidak dianalisis untuk aktivitas akses yang tidak digunakan dan peran tersebut tidak termasuk dalam jumlah total peran IAM yang dianalisis.

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 biaya dan harga IAM Access Analyzer, lihat harga IAM [Access Analyzer](#).

Untuk melihat tagihan Anda, buka Dasbor Manajemen Penagihan dan Biaya di [konsol AWS Billing and Cost Management](#). Tagihan Anda berisi tautan ke laporan penggunaan yang memberikan detail tentang tagihan Anda. Untuk mempelajari selengkapnya tentang Akun AWS penagihan, lihat [AWS Billing Panduan Pengguna](#)

Jika Anda memiliki pertanyaan tentang AWS penagihan, akun, dan acara, [hubungi AWS Support](#).

Temuan untuk akses eksternal dan tidak terpakai

IAM Access Analyzer menghasilkan temuan untuk akses eksternal dan akses yang tidak digunakan di organisasi atau 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.

IAM Access Analyzer juga menghasilkan temuan untuk akses yang tidak terpakai yang diberikan di AWS organisasi dan akun Anda. 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 memungkinkan mereka mengakses Anda. Akun AWS

- 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 dan biaya untuk temuan akses yang tidak digunakan berdasarkan jumlah peran IAM dan pengguna yang dianalisis per penganalisis per bulan. Untuk detail selengkapnya tentang harga, lihat [harga IAM Access Analyzer](#).

Topik

- [Cara kerja temuan IAM Access Analyzer](#)
- [Memulai dengan AWS Identity and Access Management Access Analyzer temuan](#)
- [Melihat dasbor temuan IAM Access Analyzer](#)
- [Bekerja dengan temuan](#)
- [Meninjau temuan](#)
- [Memfilter temuan](#)
- [Mengarsipkan temuan](#)
- [Menyelesaikan temuan](#)
- [Jenis sumber daya IAM Access Analyzer untuk akses eksternal](#)
- [Pengaturan untuk IAM Access Analyzer](#)
- [Aturan arsip](#)
- [Pemantauan AWS Identity and Access Management Access Analyzer dengan Amazon EventBridge](#)
- [Integrasikan Access Analyzer dengan AWS Security Hub](#)
- [Mencatat panggilan API IAM Access Analyzer dengan AWS CloudTrail](#)
- [Kunci filter IAM Access Analyzer](#)

- [Mengggunakan peran terkait layanan untuk AWS Identity and Access Management Access Analyzer](#)

Cara kerja temuan IAM Access Analyzer

Topik ini menjelaskan konsep dan istilah yang digunakan dalam IAM Access Analyzer untuk membantu Anda menjadi akrab dengan cara IAM Access Analyzer memantau akses ke sumber daya Anda. AWS

Akses eksternal

Untuk analisis akses eksternal, AWS Identity and Access Management Access Analyzer dibangun di atas [Zelkova](#), yang menerjemahkan kebijakan IAM ke dalam pernyataan logis yang setara, dan menjalankan serangkaian pemecah logis tujuan umum dan khusus (teori modulo kepuasan) terhadap masalah. IAM Access Analyzer menerapkan Zelkova berulang kali pada kebijakan dengan kueri yang semakin spesifik untuk mengkarakterisasi kelas perilaku yang diizinkan kebijakan, berdasarkan konten kebijakan. Untuk mempelajari lebih lanjut tentang teori modulo, lihat [Teori Modulo Kepuasan](#).

Untuk penganalisis akses eksternal, IAM Access Analyzer tidak memeriksa log akses untuk menentukan apakah entitas eksternal mengakses sumber daya dalam zona kepercayaan Anda. Ia menghasilkan temuan ketika kebijakan berbasis sumber daya memungkinkan akses ke sumber daya, bahkan jika sumber daya tidak diakses oleh entitas eksternal. IAM Access Analyzer juga tidak mempertimbangkan keadaan akun eksternal apa pun saat membuatnya. Artinya, jika ini menunjukkan bahwa akun 111122223333 dapat mengakses bucket Amazon S3 Anda, ia tidak tahu apa-apa tentang status pengguna, peran, kebijakan kontrol layanan (SCP), dan konfigurasi relevan lainnya di akun itu. Ini untuk privasi pelanggan - IAM Access Analyzer tidak mempertimbangkan siapa yang memiliki akun lain. Ini juga untuk keamanan - jika akun tidak dimiliki oleh pelanggan IAM Access Analyzer, masih penting untuk mengetahui bahwa entitas eksternal dapat memperoleh akses ke sumber daya mereka bahkan jika saat ini tidak ada prinsip dalam akun yang dapat mengakses sumber daya.

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 oleh IAM Access Analyzer, lihat Kunci filter [IAM Access Analyzer](#).

IAM Access Analyzer saat ini tidak melaporkan temuan dari kepala AWS layanan atau akun layanan internal. Dalam kasus yang jarang terjadi di mana IAM Access Analyzer tidak dapat sepenuhnya menentukan apakah pernyataan kebijakan memberikan akses ke entitas eksternal, itu salah di

sisi mendeklarasikan temuan positif palsu. IAM Access Analyzer dirancang untuk memberikan pandangan komprehensif tentang berbagi sumber daya di akun Anda, dan berusaha untuk meminimalkan negatif palsu.

Akses yang tidak digunakan

Anda harus membuat analyzer untuk temuan akses yang tidak terpakai untuk peran Anda bahkan jika Anda telah membuat analyzer untuk menghasilkan temuan akses eksternal untuk sumber daya Anda. Setelah membuat analyzer, IAM Access Analyzer meninjau aktivitas akses untuk mengidentifikasi akses yang tidak digunakan. IAM Access Analyzer meninjau informasi yang diakses terakhir untuk semua peran, kunci akses pengguna, dan kata sandi pengguna di AWS organisasi dan akun Anda untuk membantu Anda mengidentifikasi akses yang tidak digunakan. Untuk peran dan pengguna IAM aktif, IAM Access Analyzer menggunakan layanan IAM dan informasi yang terakhir diakses tindakan untuk mengidentifikasi izin yang tidak digunakan. Anda dapat menggunakan penganalisis akses yang tidak terpakai untuk menskalakan proses peninjauan Anda di tingkat AWS organisasi dan akun. Anda dapat menggunakan informasi tindakan yang terakhir diakses untuk menyelidiki lebih dalam tentang peran individu.

Dasbor ringkasan

Untuk akses eksternal dan tidak terpakai, IAM Access Analyzer mengatur temuan di dasbor ringkasan. Untuk akses eksternal, dasbor ringkasan menyoroti pemisahan antara temuan akses publik dan lintas akun, dan memberikan rincian temuan berdasarkan jenis sumber daya. Untuk akses yang tidak digunakan, dasbor menyoroti Anda Akun AWS yang memiliki temuan paling banyak dan memberikan rincian temuan berdasarkan jenis. Setelah Anda membuat penganalisis untuk akses eksternal atau tidak terpakai, IAM Access Analyzer secara otomatis menambahkan temuan baru ke dasbor yang berfokus pada peran dengan izin yang tidak digunakan.

Memulai dengan AWS Identity and Access Management Access Analyzer temuan

Gunakan informasi dalam topik ini untuk mempelajari tentang persyaratan yang diperlukan untuk menggunakan dan mengelola AWS Identity and Access Management Access Analyzer, dan kemudian cara mengaktifkan IAM Access Analyzer. Untuk mempelajari lebih lanjut tentang peran terkait layanan untuk IAM Access Analyzer, lihat [Menggunakan peran terkait layanan untuk AWS Identity and Access Management Access Analyzer](#)

Izin yang 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 kebijakan AWS terkelola untuk membantu Anda memulai dengan cepat.

- [IAM AccessAnalyzerFullAccess](#) - Memungkinkan akses penuh ke IAM Access Analyzer untuk administrator. Kebijakan ini juga memungkinkan pembuatan peran terkait layanan yang diperlukan untuk memungkinkan IAM Access Analyzer menganalisis sumber daya di akun atau organisasi Anda. AWS
- [IAM AccessAnalyzerReadOnlyAccess](#) - Memungkinkan akses hanya-baca 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](#) dalam Referensi Otorisasi Layanan.

Izin layanan IAM Access Analyzer yang diperlukan

IAM Access Analyzer menggunakan peran terkait layanan (SLR) bernama.

`AWSServiceRoleForAccessAnalyzer` SLR ini memberikan layanan akses hanya-baca 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 analisa akses eksternal dengan akun Anda sebagai zona kepercayaan.
- Anda membuat penganalisis akses yang tidak terpakai dengan akun Anda sebagai akun yang dipilih.

Untuk informasi selengkapnya, lihat [Menggunakan peran terkait layanan untuk AWS Identity and Access Management Access Analyzer](#).

Note

IAM Access Analyzer adalah Regional. Untuk akses eksternal, Anda harus mengaktifkan IAM Access Analyzer di setiap Wilayah secara independen.

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.

Dalam beberapa kasus, setelah Anda membuat akses eksternal atau penganalisis akses yang tidak digunakan di IAM Access Analyzer, halaman atau dasbor temuan 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 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 akan menghasilkan temuan.

Note

Untuk penganalisis akses eksternal, mungkin diperlukan waktu hingga 30 menit setelah kebijakan dimodifikasi untuk IAM Access Analyzer untuk menganalisis sumber daya dan kemudian menghasilkan temuan akses eksternal baru atau memperbarui temuan yang ada untuk akses ke sumber daya. Untuk penganalisis akses eksternal dan tidak terpakai, pembaruan untuk temuan mungkin tidak segera tercermin di dasbor.

Izin Penganalisis Akses IAM yang diperlukan untuk melihat dasbor temuan

Untuk melihat [dasbor temuan IAM Access Analyzer](#), 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 IAM Access Analyzer](#) di Referensi Otorisasi Layanan.

Mengaktifkan IAM Access Analyzer

Untuk membuat analisa akses eksternal dengan Akun AWS sebagai zona kepercayaan

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.

1. Buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Pilih Penganalisis akses.
3. Pilih Pengaturan Analyzer.
4. Pilih Membuat penganalisis.
5. Di bagian Analisis, pilih Analisis akses eksternal.
6. Di bagian Detail Analyzer, konfirmasi bahwa Wilayah yang ditampilkan adalah Wilayah tempat Anda ingin mengaktifkan IAM Access Analyzer.
7. Masukkan nama penganalisis.
8. Pilih Current Akun AWS sebagai zona kepercayaan untuk analyzer.

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.

9. Opsional. Tambahkan tanda yang ingin Anda terapkan ke penganalisis.
10. Pilih Kirim.

Saat Anda membuat penganalisis akses eksternal untuk mengaktifkan IAM Access Analyzer, nama peran terkait layanan akan `AWSServiceRoleForAccessAnalyzer` dibuat di akun Anda.

Untuk membuat analisa akses eksternal dengan organisasi sebagai zona kepercayaan

1. Buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Pilih Penganalisis akses.

3. Pilih Pengaturan Analyzer.
4. Pilih Membuat penganalisis.
5. Di bagian Analisis, pilih Analisis akses eksternal.
6. Di bagian Detail Analyzer, konfirmasikan bahwa Wilayah yang ditampilkan adalah Wilayah tempat Anda ingin mengaktifkan IAM Access Analyzer.
7. Masukkan nama penganalisis.
8. Pilih Organisasi saat ini sebagai zona kepercayaan untuk penganalisis.
9. Opsional. Tambahkan tanda yang ingin Anda terapkan ke penganalisis.
10. 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.

Untuk membuat 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](#).

1. Buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Pilih Penganalisis akses.
3. Pilih Pengaturan Analyzer.
4. Pilih Membuat penganalisis.
5. Di bagian Analisis, pilih Analisis akses yang tidak digunakan.
6. Masukkan nama penganalisis.
7. Untuk periode Tracking, masukkan jumlah hari untuk menghasilkan temuan untuk izin yang tidak digunakan. Misalnya, jika Anda memasukkan 90 hari, penganalisis akan menghasilkan temuan untuk entitas IAM dalam akun yang dipilih untuk setiap izin yang belum digunakan dalam 90 hari atau lebih sejak pemindaian terakhir penganalisis. Anda dapat memilih nilai antara 1 dan 180 hari.

8. Untuk Akun yang Dipilih, pilih Saat Ini Akun AWS.

 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. Opsional. Tambahkan tanda yang ingin Anda terapkan ke penganalisis.


10. Pilih Kirim.

Saat Anda membuat penganalisis akses yang tidak digunakan untuk mengaktifkan IAM Access Analyzer, peran terkait layanan bernama akan `AWSServiceRoleForAccessAnalyzer` dibuat di akun Anda.

Untuk membuat penganalisis akses yang tidak terpakai 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. Pilih Penganalisis akses.
3. Pilih Pengaturan Analyzer.

4. Pilih Membuat penganalisis.
5. Di bagian Analisis, pilih Analisis akses yang tidak digunakan.
6. Masukkan nama penganalisis.
7. Untuk periode Tracking, masukkan jumlah hari untuk menghasilkan temuan untuk izin yang tidak digunakan. Misalnya, jika Anda memasukkan 90 hari, penganalisis akan menghasilkan temuan untuk entitas IAM dalam akun organisasi yang dipilih untuk izin apa pun yang belum digunakan dalam 90 hari atau lebih sejak pemindaian terakhir penganalisis. Anda dapat memilih nilai antara 1 dan 180 hari.
8. Untuk Akun yang dipilih, pilih Organisasi saat ini sebagai akun yang dipilih untuk penganalisis.
9. Opsional. Tambahkan tanda yang ingin Anda terapkan ke penganalisis.
10. Pilih Kirim.

Saat Anda membuat penganalisis akses yang tidak digunakan untuk mengaktifkan IAM Access Analyzer, peran terkait layanan bernama akan `AWSServiceRoleForAccessAnalyzer` dibuat di akun Anda.

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	<p>Untuk penganalisis akses eksternal, penganalisis secara aktif memantau sumber daya dalam zona kepercayaannya. Analyzer secara aktif menghasilkan temuan baru dan memperbarui temuan yang ada.</p> <p>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.</p>

Status	Deskripsi
Membuat	Pembuatan penganalisis masih dalam proses. Penganalisis menjadi aktif setelah pembuatan selesai.
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.
Failed	Pembuatan penganalisis gagal karena masalah konfigurasi. Penganalisis tidak akan menghasilkan temuan apa pun. Hapus penganalisis dan buat penganalisis baru.

Melihat dasbor temuan IAM Access Analyzer

AWS Identity and Access Management Access Analyzer mengatur akses eksternal dan temuan akses yang tidak digunakan ke dalam dasbor ringkasan visual. Dasbor membantu Anda mendapatkan visibilitas ke dalam penggunaan izin yang efektif dalam skala besar dan mengidentifikasi akun yang perlu diperhatikan. Anda dapat menggunakan dasbor untuk meninjau temuan berdasarkan AWS organisasi, akun, dan jenis temuan.

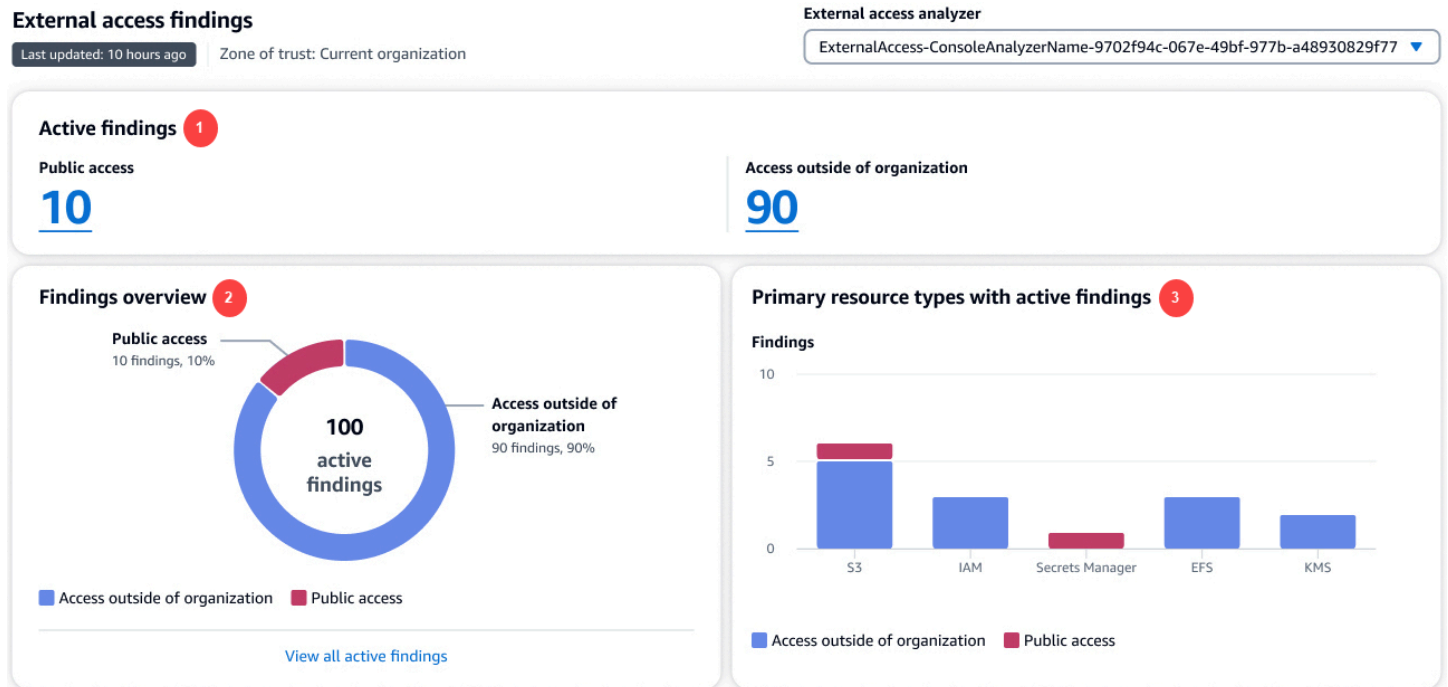
Untuk melihat dasbor ringkasan untuk penganalisis akses eksternal

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/>.

- Pilih Penganalisis akses. Jendela Ringkasan ditampilkan.
- Pilih analyzer dari dropdown External access analyzer. Ringkasan temuan untuk penganalisis ditampilkan di bagian Temuan akses eksternal.




Pada gambar sebelumnya, dasbor temuan akses eksternal terlihat dari dalam halaman Ringkasan:

- Bagian temuan aktif mencakup jumlah temuan aktif untuk akses publik dan jumlah temuan aktif yang menyediakan akses di luar akun atau organisasi. Pilih nomor untuk mencantumkan semua temuan aktif dari setiap jenis.
- Bagian ikhtisar Temuan mencakup rincian jenis temuan aktif. Pilih Lihat semua temuan aktif untuk daftar lengkap temuan aktif untuk akun atau organisasi penganalisis.
- Jenis sumber daya primer dengan bagian temuan aktif mencakup rincian jenis sumber daya primer dengan temuan aktif. Informasi ini membantu Anda memprioritaskan temuan untuk sumber daya utama terlebih dahulu. Misalnya, Amazon S3, DynamoDB, dan AWS KMS Ini bukan daftar lengkap dari setiap jenis sumber daya. Penganalisis Anda mungkin memiliki temuan aktif untuk jenis sumber daya yang tidak tercantum di bagian ini.

Untuk melihat dasbor ringkasan untuk 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. Untuk detail selengkapnya tentang harga, lihat harga [IAM Access Analyzer](#).

 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 Penganalisis akses. Jendela Ringkasan ditampilkan.
3. Pilih penganalisis dari dropdown penganalisis akses yang tidak digunakan. Ringkasan temuan untuk penganalisis ditampilkan di bagian Temuan akses yang tidak digunakan.

Unused access findings

Unused access analyzer

Last updated: 10 hours ago

Tracking period: 90 days

Current organization

UsedAccess-ConsoleAnalyzerName-9702f94c-067e-49bf-977b-a48930829f77

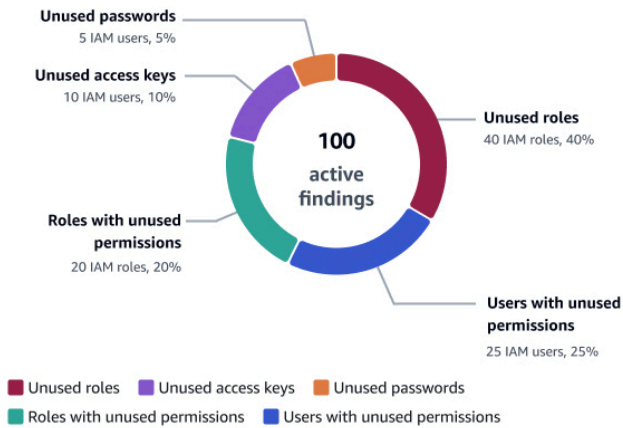
Active findings 1

Unused roles
40

Unused credentials
15

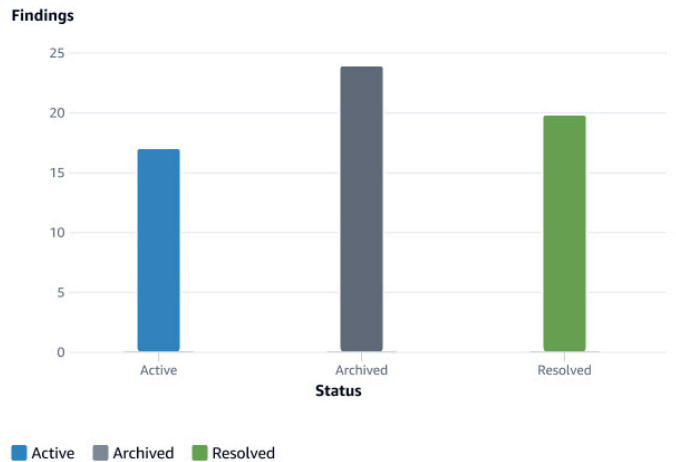
Unused permissions
45

Findings overview 2



[View all active findings](#)

Finding status 3



Accounts with the most findings for unused access 4

Account	Active findings	Findings by type
Audit 11111111111111	15	Unused roles, Unused access keys, Unused passwords, Roles with unused permissions, Users with unused permissions
Log 22222222222222	10	Unused roles, Unused access keys, Unused passwords, Roles with unused permissions, Users with unused permissions
Security 33333333333333	10	Unused roles, Unused access keys, Unused passwords, Roles with unused permissions, Users with unused permissions
Production 44444444444444	10	Unused roles, Unused access keys, Unused passwords
Sandbox 55555555555555	5	Unused access keys, Roles with unused permissions, Users with unused permissions

Pada gambar sebelumnya, dasbor temuan akses eksternal 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.
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.

Bekerja dengan temuan

Temuan akses eksternal

Temuan akses eksternal dihasilkan hanya sekali untuk setiap contoh sumber daya yang dibagikan di luar zona kepercayaan Anda. Setiap kali kebijakan berbasis sumber daya diubah, 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. Jika akses dalam temuan pertama dihapus, temuan itu diperbarui ke status Terselesaikan.

Status semua temuan tetap Aktif sampai Anda mengarsipkannya atau menghapus akses yang menghasilkan temuan. Saat Anda menghapus akses, status temuan diperbarui ke Terselesaikan.

Note

Diperlukan waktu hingga 30 menit setelah kebijakan diubah untuk IAM Access Analyzer untuk menganalisis sumber daya dan kemudian memperbarui temuan akses eksternal.

Temuan akses yang tidak digunakan

Temuan akses yang tidak digunakan dihasilkan untuk entitas IAM 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 dari 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.

Anda harus meninjau semua temuan di akun Anda untuk menentukan apakah akses eksternal atau tidak terpakai diharapkan dan disetujui. Jika akses eksternal atau tidak terpakai 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.

Note

Temuan akses yang tidak digunakan hanya tersedia menggunakan aksi API [ListFindingsV2](#).

Meninjau temuan

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.

Untuk meninjau temuan

1. Buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Pilih Penganalisis akses.
3. Dasbor temuan ditampilkan. Pilih temuan aktif untuk penganalisis akses eksternal atau tidak terpakai Anda.


Untuk informasi lebih lanjut tentang melihat dasbor temuan, lihat [Melihat dasbor temuan IAM Access Analyzer](#).

 Note

Temuan ditampilkan hanya jika Anda memiliki izin untuk melihat temuan untuk penganalisis.

Semua temuan ditampilkan untuk penganalisis. Untuk melihat temuan lain yang dihasilkan oleh penganalisis, pilih jenis temuan yang sesuai dari tarik-turun Status:

- Pilih Aktif untuk melihat semua temuan aktif yang dihasilkan oleh penganalisis.
- Pilih Diarsipkan untuk melihat hanya temuan yang dihasilkan oleh penganalisis yang telah diarsipkan. Untuk mempelajari informasi lebih lanjut, lihat [Mengarsipkan temuan](#).
- Pilih Diselesaikan untuk melihat hanya temuan yang dihasilkan oleh penganalisis yang telah diselesaikan. Saat Anda memulihkan masalah yang menghasilkan temuan, status temuan diubah menjadi Terselesaikan.

 Important

Temuan yang diselesaikan dihapus 90 hari setelah pembaruan terakhir temuan. Temuan aktif dan diarsipkan tidak akan dihapus kecuali jika Anda menghapus penganalisis yang menghasilkannya.

- Pilih Semua untuk melihat semua temuan dengan semua status yang dihasilkan oleh penganalisis.

Temuan akses eksternal

Pilih Akses eksternal dan kemudian pilih penganalisis akses eksternal dari menu tarik-turun View analyzer. Halaman Temuan untuk penganalisis akses eksternal menampilkan detail berikut tentang sumber daya bersama dan pernyataan kebijakan yang menghasilkan temuan:

Menemukan ID

ID unik yang ditetapkan untuk temuan. Pilih ID temuan untuk menampilkan detail tambahan tentang sumber daya dan pernyataan kebijakan yang menghasilkan temuan tersebut.

Sumber

Jenis dan nama parsial dari sumber daya yang memiliki kebijakan yang diterapkan yang memberikan akses ke entitas eksternal yang tidak berada di zona kepercayaan Anda.

Akun pemilik sumber daya

Kolom ini ditampilkan hanya jika Anda menggunakan sebuah organisasi sebagai zona kepercayaan. Akun dalam organisasi yang memiliki sumber daya yang dilaporkan dalam temuan.

Prinsipal eksternal

Penanggung jawab, tidak di dalam zona kepercayaan Anda yang diberikan akses oleh kebijakan yang dianalisis. Nilai yang valid meliputi:

- Akun AWS— Semua prinsipal dalam daftar Akun AWS dengan izin dari administrator akun itu dapat mengakses sumber daya.
- Prinsipal apa pun — Semua kepala sekolah Akun AWS yang memenuhi persyaratan yang termasuk dalam kolom Ketentuan memiliki izin untuk mengakses sumber daya. Misalnya, jika VPC dicantumkan, artinya setiap penanggung jawab dalam akun apa pun yang memiliki izin untuk mengakses VPC yang tercantum dapat mengakses sumber daya.
- Pengguna kanonik — Semua kepala sekolah Akun AWS dengan ID pengguna kanonik yang terdaftar memiliki izin untuk mengakses sumber daya.
- Peran IAM – Peran IAM yang terdaftar memiliki izin untuk mengakses sumber daya.
- Pengguna IAM – Pengguna IAM yang tercantum memiliki izin untuk mengakses sumber daya.

Kondisi

Ketentuan dari pernyataan kebijakan yang memberikan akses. Misalnya, jika kolom Ketentuan mencakup VPC Sumber, artinya sumber daya dibagikan dengan penanggung jawab yang memiliki akses ke VPC yang tercantum. Ketentuan dapat bersifat global atau khusus layanan.

[Kunci ketentuan global](#) memiliki aws : awalan.

Dibagikan melalui

Bidang Dibagikan melalui menunjukkan bagaimana akses yang menghasilkan temuan diberikan.

Nilai yang valid meliputi:

- Kebijakan Bucket — Kebijakan bucket yang dilampirkan pada bucket Amazon S3.
- Daftar kontrol akses — Daftar kontrol akses (ACL) yang dilampirkan ke bucket Amazon S3.
- Titik akses — Titik akses atau titik akses multi-wilayah yang terkait dengan bucket Amazon S3. ARN titik akses ditampilkan dalam detail Temuan.

Tingkat akses

Tingkat akses yang diberikan kepada entitas eksternal oleh tindakan dalam kebijakan berbasis sumber daya. Lihat detail temuan untuk informasi lebih lanjut. Nilai tingkat akses mencakup hal berikut:

- 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.
- Baca – Izin untuk membaca, namun tidak mengedit konten dan atribut sumber daya dalam layanan.
- Tulis – Izin untuk membuat, menghapus, atau memodifikasi sumber daya dalam layanan.
- Izin – Izin untuk memberikan atau mengubah izin sumber daya dalam layanan.
- Penandaan – Izin untuk melakukan tindakan yang hanya mengubah status tanda sumber daya.

Diperbarui

Stempel waktu untuk pembaruan terbaru ke status temuan, atau waktu dan tanggal temuan dihasilkan jika tidak ada pembaruan yang dilakukan.

Note

Diperlukan waktu hingga 30 menit setelah kebijakan dimodifikasi untuk IAM Access Analyzer untuk menganalisis kembali sumber daya dan kemudian memperbarui temuannya.

Status

Status temuan, salah satu Aktif, Diarsipkan, atau Diselesaikan.

Temuan 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. Untuk detail selengkapnya tentang harga, lihat harga [IAM Access Analyzer](#).

Pilih Akses yang tidak digunakan dan kemudian pilih penganalisis akses yang tidak terpakai dari menu tarik-turun View analyzer. Halaman Temuan untuk penganalisis akses yang tidak digunakan menampilkan detail berikut tentang entitas IAM yang menghasilkan temuan:

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).

Memfilter temuan

Pemfilteran default untuk halaman temuan adalah untuk menampilkan semua temuan. Untuk melihat temuan aktif, pilih status Aktif dari tarik-turun Status. Untuk melihat temuan yang diarsipkan, pilih Status yang diarsipkan dari menu tarik-turun Status. 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. Misalnya, untuk membuat filter yang hanya menampilkan temuan tertentu Akun AWS, pilih AWS Akun untuk properti, lalu pilih AWS Akun =, lalu masukkan nomor akun Akun AWS yang ingin Anda lihat temuannya.

Untuk daftar kunci filter yang dapat Anda gunakan untuk membuat atau memperbarui aturan arsip, lihat [Kunci filter IAM Access Analyzer](#).

Memfilter temuan akses eksternal

Untuk menyaring temuan akses eksternal

1. Pilih Akses eksternal dan kemudian pilih analyzer di menu tarik-turun View analyzer.
2. Pilih kotak pencarian untuk menampilkan daftar properti yang tersedia.
3. Pilih properti yang akan digunakan untuk memfilter temuan yang ditampilkan.
4. 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 menentukan filter:

- 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.

- **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.
- **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](#).

- 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 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 berdasarkan 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](#).
- 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.
- 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.

Memfilter temuan akses yang tidak digunakan

Untuk menyaring temuan akses yang tidak digunakan

1. Pilih Akses yang tidak digunakan dan kemudian pilih penganalisis di menu tarik-turun View analyzer.
2. Pilih kotak pencarian untuk menampilkan daftar properti yang tersedia.
3. Pilih properti yang akan digunakan untuk memfilter temuan yang ditampilkan.
4. 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 = UnuseDiamRole, Hanya temuan dengan tipe UnuseDiamRole 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.

Mengarsipkan temuan

Ketika Anda mendapatkan temuan untuk akses ke sumber daya yang disengaja, Anda dapat mengarsipkan temuan. Misalnya, pencarian akses eksternal untuk peran IAM yang digunakan oleh beberapa pengguna 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 Batal Arsip alih-alih Arsipkan. Saat Anda menghapus arsip temuan, status diatur ke Aktif.

Menyelesaikan temuan

Temuan akses eksternal

Untuk menyelesaikan temuan akses eksternal yang dihasilkan dari akses yang tidak ingin Anda izinkan, ubah pernyataan kebijakan untuk menghapus izin yang memungkinkan akses ke sumber daya yang diidentifikasi. Misalnya, untuk temuan di 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. Gunakan konsol untuk sumber daya lain yang didukung untuk memodifikasi pernyataan kebijakan yang menghasilkan temuan yang dihasilkan.

Setelah Anda membuat perubahan untuk menyelesaikan pencarian akses eksternal, seperti memodifikasi kebijakan yang diterapkan ke peran IAM, IAM Access Analyzer memindai kembali sumber daya tersebut. Jika sumber daya tidak lagi dibagikan di luar zona kepercayaan Anda, status temuan diubah menjadi Diselesaikan. Temuan ini tidak lagi ditampilkan dalam daftar temuan aktif, dan sebagai gantinya ditampilkan dalam daftar temuan yang diselesaikan.

Note

Ini tidak berlaku untuk temuan Kesalahan. Ketika IAM Access Analyzer tidak dapat menganalisis sumber daya, itu menghasilkan temuan kesalahan. Jika Anda mengatasi masalah yang mencegah IAM Access Analyzer menganalisis sumber daya, temuan kesalahan akan dihapus sepenuhnya daripada berubah menjadi temuan yang diselesaikan.

Jika perubahan yang Anda buat mengakibatkan sumber daya dibagikan di luar zona kepercayaan Anda, tetapi dengan cara yang berbeda, seperti dengan prinsipal yang berbeda atau untuk izin yang berbeda, IAM Access Analyzer menghasilkan temuan Aktif baru.

Note

Diperlukan waktu hingga 30 menit setelah kebijakan dimodifikasi untuk IAM Access Analyzer untuk menganalisis kembali sumber daya dan kemudian memperbarui temuannya. Temuan yang diselesaikan dihapus 90 hari setelah pembaruan terakhir ke status temuan.

Temuan akses yang tidak digunakan

Untuk mengatasi temuan akses yang tidak digunakan, gunakan konsol IAM untuk menghapus kunci akses, kata sandi, izin, atau peran yang tidak digunakan. Untuk informasi selengkapnya, lihat sumber daya berikut:

- Untuk informasi selengkapnya tentang menghapus kunci akses, lihat [Mengelola kunci akses \(konsol\)](#).
- Untuk informasi selengkapnya tentang menghapus kata sandi pengguna IAM, lihat [Membuat, mengubah, atau menghapus kata sandi pengguna \(konsol\) IAM](#).

- Untuk informasi selengkapnya tentang mengubah izin untuk pengguna IAM, lihat [Mengubah izin untuk pengguna \(konsol\)](#).
- Untuk informasi selengkapnya tentang menghapus peran IAM, lihat [Menghapus peran IAM \(konsol\)](#).

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, Anda hanya menghapus beberapa izin yang tidak digunakan dalam sebuah 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](#).

Jenis sumber daya IAM Access Analyzer untuk akses eksternal

Untuk penganalisis akses eksternal, 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. Tinjau informasi tentang setiap sumber daya untuk detail tentang bagaimana IAM Access Analyzer menghasilkan temuan untuk setiap jenis sumber daya.

Note

Jenis sumber daya yang didukung yang tercantum adalah untuk penganalisis akses eksternal. Penganalisis akses yang tidak digunakan hanya mendukung pengguna dan peran IAM. Untuk informasi selengkapnya, lihat [Bekerja dengan temuan](#).

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](#)

Bucket Amazon Simple Storage Service

Saat IAM Access Analyzer menganalisis bucket Amazon S3, ia akan menghasilkan temuan saat kebijakan bucket Amazon S3, 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.

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.

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 menggunakan kelas penyimpanan Amazon S3 Express One, yang direkomendasikan untuk beban kerja atau aplikasi yang kritis terhadap kinerja. Untuk bucket direktori Amazon S3, IAM Access Analyzer menganalisis kebijakan bucket direktori, termasuk pernyataan kondisi dalam kebijakan, yang memungkinkan entitas eksternal mengakses bucket direktori. Untuk informasi selengkapnya tentang bucket direktori Amazon S3, lihat 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 dienkrpsi. 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 dienkrpsi, 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 snapshot Amazon RDS DB, IAM Access Analyzer menganalisis daftar kontrol akses yang memungkinkan entitas eksternal mengakses snapshot. 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.

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 snapshot klaster Amazon RDS DB, IAM Access Analyzer menganalisis daftar kontrol akses yang memungkinkan entitas eksternal mengakses snapshot. Snapshot cluster yang tidak terenkripsi dapat bersifat publik. Snapshot klaster 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.

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 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

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](#).

Tabel Amazon DynamoDB

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](#).

Pengaturan 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

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 didelegasikan. 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 organisasi AWS, 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 pada 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](#).

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 dengan menggunakan AWS CLI atau SDK AWS

Saat Anda membuat penganalisis untuk akses penganalisis di seluruh organisasi dalam akun administrator yang didelegasikan menggunakan AWS CLI, AWS API (menggunakan AWS SDK) atau AWS CloudFormation, Anda harus menggunakan AWS Organizations API 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 Pengguna AWS Organizations.

2. Daftarkan akun anggota organisasi AWS Anda yang valid sebagai administrator yang didelegasikan menggunakan operasi API AWS Organizations [RegisterDelegatedAdministrator](#) atau perintah `register-delegated-administrator` AWS CLI.

Setelah Anda mengubah administrator yang didelegasikan, administrator yang baru harus membuat penganalisis untuk mulai memantau akses ke sumber daya di organisasi Anda.

Menghapus penganalisis

Anda dapat menghapus penganalisis akses eksternal dan tidak terpakai yang ada dari halaman pengaturan Analyzer. Saat Anda menghapus penganalisis, sumber daya yang ditentukan dalam penganalisis tidak lagi dipantau dan tidak ada temuan baru yang dihasilkan. Semua temuan yang dihasilkan oleh penganalisis dihapus.

Untuk temuan yang dihapus karena penganalisis yang menghasilkannya dihapus, acara dikirim ke EventBridge dalam dua hari berikutnya setelah penganalisis dihapus. Diperlukan waktu hingga 90 hari setelah penganalisis dihapus agar temuan Security Hub dihapus.


Untuk menghapus penganalisis

1. Buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Di bawah Access Analyzer, pilih Pengaturan Analyzer.
3. Pilih analyzer yang akan dihapus dan kemudian pilih Delete.
4. Ketik **delete** kotak teks konfirmasi dan kemudian pilih Hapus.

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 Delete (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.

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 peng analisis 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::my-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 ke 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:::my-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 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 ke 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:::example-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:::example-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::example-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 akses eksternal atau peristiwa temuan akses yang tidak digunakan, gunakan contoh pola berikut:

```
{
  "source": [
    "aws.access-analyzer"
  ],
  "detail-type": [
    "Access Analyzer 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"
  ]
}
```

Note

Anda tidak dapat membuat aturan hanya berdasarkan peristiwa temuan akses eksternal.

- 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 AWSlayanan, 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 Access Analyzer dengan AWS Security Hub

[AWS Security Hub](#) memberi Anda pandangan komprehensif tentang keadaan keamanan Anda AWS dan membantu Anda memeriksa lingkungan Anda terhadap standar industri keamanan dan praktik terbaik. Security Hub mengumpulkan data keamanan dari berbagai AWS akun, layanan, dan produk mitra pihak ketiga yang didukung serta membantu Anda menganalisis tren keamanan dan mengidentifikasi masalah keamanan prioritas tertinggi.

Saat Anda berintegrasi AWS Identity and Access Management Access Analyzer dengan Security Hub, Anda dapat mengirim temuan dari IAM Access Analyzer ke Security Hub. Security Hub kemudian dapat menyertakan temuan tersebut dalam analisis postur keamanan Anda.

Daftar Isi

- [Bagaimana IAM Access Analyzer mengirimkan temuan ke Security Hub](#)
 - [Jenis temuan yang dikirim IAM Access Analyzer](#)
 - [Latensi untuk mengirim temuan](#)
 - [Mencoba kembali saat Security Hub tidak tersedia](#)
 - [Memperbarui temuan yang ada di Security Hub](#)
- [Melihat temuan IAM Access Analyzer di Security Hub](#)
 - [Menafsirkan IAM Access Analyzer menemukan nama di Security Hub](#)
- [Temuan khas dari IAM Access Analyzer](#)
- [Mengaktifkan dan mengonfigurasi integrasi](#)
- [Bagaimana cara menghentikan pengiriman temuan](#)

Bagaimana IAM Access Analyzer mengirimkan temuan ke Security Hub

Di Security Hub, masalah keamanan dilacak sebagai temuan. Beberapa temuan berasal dari masalah yang terdeteksi oleh AWS layanan lain atau oleh mitra pihak ketiga. Security Hub juga memiliki seperangkat aturan yang digunakan untuk mendeteksi masalah keamanan dan menghasilkan temuan.

Security Hub menyediakan alat untuk mengelola temuan dari seluruh sumber tersebut. Anda dapat melihat dan memfilter daftar temuan dan melihat detail untuk temuan. Lihat [Melihat temuan](#) di Panduan Pengguna AWS Security Hub . Anda juga dapat melacak status penyelidikan ke temuan. Lihat [Mengambil tindakan pada temuan](#) di Panduan Pengguna AWS Security Hub .

Semua temuan di Security Hub menggunakan format JSON standar yang disebut AWS Security Finding Format (ASFF). ASFF mencakup detail tentang sumber masalah, sumber daya yang terdampak, dan status temuan saat ini. Lihat [AWS Security Finding Format \(ASFF\)](#) di Panduan Pengguna AWS Security Hub .

AWS Identity and Access Management Access Analyzer adalah salah satu AWS layanan yang mengirimkan temuan ke Security Hub. 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. 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, yang kemudian dikirim ke Security Hub. Untuk akses lintas akun, IAM Access Analyzer mengirimkan satu temuan untuk satu prinsipal eksternal sekaligus ke Security Hub. Jika ada beberapa temuan lintas akun di IAM Access Analyzer, Anda harus menyelesaikan temuan 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 peng analisis, Anda harus melihat temuan di IAM Access Analyzer.

Jenis temuan yang dikirim IAM Access Analyzer

IAM Access Analyzer mengirimkan temuan ke Security Hub 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 - Efek/Paparan Data/Akses Eksternal Diberikan

- Temuan akses eksternal — Pemeriksaan Perangkat Lunak dan Konfigurasi/Praktik Terbaik AWS Keamanan/Akses Eksternal Diberikan
- Temuan akses yang tidak digunakan — Pemeriksaan Perangkat Lunak dan Konfigurasi/Praktik Terbaik AWS Keamanan/Izin yang Tidak Digunakan
- Temuan akses yang tidak digunakan — Pemeriksaan Perangkat Lunak dan Konfigurasi/Praktik Terbaik AWS Keamanan/Peran IAM yang Tidak Digunakan
- Temuan akses yang tidak digunakan — Pemeriksaan Perangkat Lunak dan Konfigurasi/Praktik Terbaik AWS Keamanan/Kata Sandi Pengguna IAM yang Tidak Digunakan
- Temuan akses yang tidak digunakan — Pemeriksaan Perangkat Lunak dan Konfigurasi/Praktik Terbaik AWS Keamanan/Kunci Akses Pengguna IAM yang Tidak Digunakan

Latensi untuk mengirim temuan

Ketika IAM Access Analyzer membuat temuan baru, biasanya dikirim ke Security Hub dalam waktu 30 menit. Jarang, dan dalam kondisi tertentu, IAM Access Analyzer tidak diberi tahu bahwa kebijakan telah ditambahkan atau diperbarui. Misalnya, perubahan ke pengaturan akses publik blok akun Amazon S3 dapat memakan waktu hingga 12 jam. Selain itu, jika ada masalah pengiriman dengan pengiriman AWS CloudTrail log, perubahan kebijakan tidak memicu pemindaian ulang sumber daya yang dilaporkan dalam temuan tersebut. Ketika ini terjadi, IAM Access Analyzer menganalisis kebijakan baru atau yang diperbarui selama pemindaian berkala berikutnya.

Mencoba kembali saat Security Hub tidak tersedia

Jika Security Hub tidak tersedia, IAM Access Analyzer mencoba mengirimkan temuan secara berkala.

Memperbarui temuan yang ada di Security Hub

Setelah mengirimkan temuan ke Security Hub, AWS Identity and Access Management Access Analyzer mengirimkan pembaruan untuk mencerminkan pengamatan tambahan dari aktivitas temuan ke Security Hub. Pembaruan tercermin dalam temuan yang sama.

Karena IAM Access Analyzer mengelompokkan temuan akses eksternal per sumber daya, temuan untuk sumber daya di Security Hub aktif jika setidaknya salah satu temuan untuk sumber daya di IAM Access Analyzer aktif. Jika semua temuan di IAM Access Analyzer untuk sumber daya diarsipkan atau diselesaikan, maka temuan Security Hub diarsipkan. Temuan Security Hub diperbarui saat Anda mengubah akses kebijakan antara akses publik dan lintas akun. Pembaruan ini dapat mencakup perubahan jenis, judul, deskripsi, dan tingkat keparahan temuan.

IAM Access Analyzer tidak mengelompokkan temuan akses yang tidak digunakan berdasarkan sumber daya, jadi jika temuan akses yang tidak digunakan diselesaikan di IAM Access Analyzer, maka temuan Security Hub diselesaikan. Temuan 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

Untuk melihat temuan IAM Access Analyzer Anda di Security Hub, 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

AWS Identity and Access Management Access Analyzer mengirimkan temuan ke Security Hub 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.

Tipe temuan ASFF	Judul pencarian Security Hub	Deskripsi
Efek/Eksposur Data/Akses Eksternal 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 Konfigurasi/Praktik Terbaik AWS Keamanan/Akses Eksternal Diberikan	<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 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 Konfigurasi/Praktik Terbaik AWS Keamanan/Peran IAM yang Tidak Digunakan	<resource ARN>berisi peran IAM yang tidak digunakan	Pengguna atau peran berisi peran IAM yang tidak digunakan.
Pemeriksaan Perangkat Lunak dan Konfigurasi/Praktik Terbaik AWS Keamanan/Kata Sandi Pengguna IAM yang Tidak Digunakan	<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 Konfigurasi/Praktik Terbaik AWS Keamanan/Kunci Akses Pengguna IAM yang Tidak Digunakan	<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 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:::my-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::my-bucket/ allows cross-account access",
    "Description": "AWS::S3::Bucket/arn:aws:s3::my-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::my-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": "DEM024d8d-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-D0-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, Anda harus mengaktifkan Security Hub. Untuk informasi tentang cara mengaktifkan Security Hub, lihat [Menyiapkan Security Hub](#) di Panduan Pengguna AWS Security Hub .

Saat Anda mengaktifkan IAM Access Analyzer dan Security Hub, integrasi diaktifkan secara otomatis. IAM Access Analyzer segera mulai mengirimkan temuan ke Security Hub.

Bagaimana cara menghentikan pengiriman temuan

Untuk berhenti mengirim temuan ke Security Hub, Anda dapat menggunakan konsol 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 denganAWS CloudTrail

IAM Access Analyzer terintegrasi denganAWS CloudTrail, layanan yang menyediakan catatan tindakan yang diambil oleh pengguna, peran, atauAWS layanan di IAM Access Analyzer. CloudTrail merekam semua panggilan API untuk IAM Access Analyzer sebagai peristiwa. Panggilan yang direkam mencakup panggilan dari konsol IAM Access Analyzer dan panggilan kode ke operasi IAM Access Analyzer.

Jika Anda membuat jejak, Anda dapat mengaktifkan pengiriman berkelanjutan dari CloudTrail peristiwa ke bucket Amazon S3, termasuk peristiwa untuk IAM Access Analyzer. Jika tidak mengonfigurasi jejak, Anda masih dapat melihat peristiwa terbaru di CloudTrail konsol diRiwayat peristiwa peristiwa.

Menggunakan informasi yang dikumpulkan oleh CloudTrail, Anda dapat menentukan permintaan yang dibuat ke IAM Access Analyzer, alamat IP tempat permintaan tersebut dibuat, siapa yang membuat permintaan, kapan permintaan dibuat, kapan permintaan dibuat, dan detail lainnya.

Untuk mempelajari selengkapnya tentang CloudTrail, lihat[AWS CloudTrailPanduan Pengguna](#).

IAM Access Analyzer di CloudTrail

CloudTrail diaktifkan pada AndaAWS akun saat Anda membuat akun. Saat aktivitas terjadi di IAM Access Analyzer, aktivitas tersebut dicatat dalam a a CloudTrail acara bersama dengan lainnyaAWS peristiwa layanan diRiwayat peristiwa peristiwa. Anda dapat melihat, mencari, dan mengunduh peristiwa terbaru di akun AWS Anda. Untuk informasi selengkapnya, lihat[Menampilkan Peristiwa dengan CloudTrail Riwayat peristiwa peristiwa](#).

Untuk catatan berkelanjutan tentang peristiwa di AndaAWS akun, termasuk peristiwa untuk IAM Access Analyzer, buat jejak. SEBUAHjejakmenyalakan CloudTrail untuk mengirimkan berkas log ke bucket Amazon S3. Secara default, saat Anda membuat jejak di dalam konsol tersebut, jejak diterapkan ke semua Wilayah AWS. Jejak mencatat peristiwa dari semua Wilayah di partisi

AWS dan mengirimkan file log ke bucket Amazon S3 yang Anda tentukan. Selain itu, Anda dapat mengonfigurasi lainnya AWS layanan untuk menganalisis lebih lanjut dan menindaklanjuti data peristiwa peristiwa yang dikumpulkan di CloudTrail log. Untuk informasi selengkapnya, lihat yang berikut:

- [Ikhtisar untuk Membuat Jejak](#)
- [CloudTrail Layanan dan Integrasi yang Didukung](#)
- [Mengonfigurasi Notifikasi Amazon SNS untuk CloudTrail](#)
- [Menerima CloudTrail Berkas log dari Beberapa Wilayah](#) dan [Menerima CloudTrail Berkas 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 di CloudTrail berkas log.

Setiap entri peristiwa atau log berisi informasi tentang siapa yang membuat permintaan tersebut. Informasi identitas membantu Anda menentukan hal berikut:

- Bahwa permintaan dibuat dengan kredensial pengguna root atau pengguna AWS Identity and Access Management (IAM).
- Bahwa permintaan tersebut dibuat dengan kredensial keamanan sementara untuk peran atau pengguna gabungan.
- Bahwa permintaan dibuat oleh layanan AWS lain.

Untuk informasi lain, lihat [Elemen userIdentity CloudTrail](#).

Memahami entri berkas log IAM Access Analyzer

Jejak adalah konfigurasi yang memungkinkan pengiriman peristiwa sebagai berkas log ke bucket Amazon S3 yang Anda tentukan. CloudTrail berkas log berisi satu atau beberapa entri log. Sebuah peristiwa mewakili permintaan tunggal dari sumber apa pun dan mencakup informasi tentang tindakan yang diminta, tanggal dan waktu tindakan, parameter permintaan, dan sebagainya. CloudTrail berkas log bukan jejak tumpukan terurut dari panggilan API publik, sehingga berkas tersebut tidak muncul dalam urutan tertentu.

Contoh berikut menunjukkan a CloudTrail entri log yang menunjukkanCreateAnalyzeroperasi yang dibuat oleh sesi peran yang diasumsikan bernamaAlice-tempcredspada tanggal "14 Juni 2021". Sesi peran dikeluarkan oleh peran bernamaadmin-tempcreds.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AROAIIBKEVSQ6C2EXAMPLE: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 AWS CloudFormation untuk mengonfigurasi aturan arsip.

Kriteria	Deskripsi	Jenis	Aturan arsip	Daftar temuan	Daftar temuan pratinjau akses
sumber daya	ARN secara unik mengidentifikasi sumber daya yang dapat diakses oleh penanggung jawab eksternal. Untuk mempelajari lebih lanjut, lihat Nama sumber daya Amazon (ARN) .	String	 Ya	 Ya	 Ya
resourceType AWS::IAM::Role	Jenis sumber daya yang dapat diakses oleh penanggung jawab eksternal.	String	 Ya	 Ya	 Ya

Kriteria	Deskripsi	Jenis	Aturan arsip	Daftar temuan	Daftar temuan pratinjau akses
AWS::KMS: :Key AWS::Lamb da::Funct ion AWS::Lamb da::Layer Version AWS::S3:: Bucket AWS::S3Ex press:: <di </di rectoryBu cket AWS::SQS: :Queue AWS::Secr etsManage r::Secret AWS::EFS: :FileSyst em AWS::EC2: :Snapshot AWS::ECR: :Reposito ry AWS::RDS: :DBSnapsh					







Kriteria	Deskripsi	Jenis	Aturan arsip	Daftar temuan	Daftar temuan pratinjau akses
ot AWS::RDS: :DBClusterSnapshot AWS::SNS: :Topic AWS::Dyna moDB::Str eam AWS::Dyna moDB::Tab le					
resourceO wnerAccou nt	12 digit ID AWS akun yang memiliki sumber daya. Untuk mempelajari lebih lanjut, lihat pengidentifikasi akun AWS .	Tali	 Ya	 Ya	 Ya
isPublic	Menunjukkan apakah temuan melaporkan sumber daya yang memiliki kebijakan yang memungkinkan akses publik.	Boolean	 Ya	 Ya	 Ya
















Kriteria	Deskripsi	Jenis	Aturan arsip	Daftar temuan	Daftar temuan pratinjau akses
FindingType UnusedIAMRole UnusedIAMUserAccessKey UnusedIAMUserPassword UnusedPermission	Jenis temuannya. Anda hanya dapat memfilter dengan menemukan jenis untuk temuan akses yang tidak digunakan.	String	 Ya	 Ya	 Ya
status ACTIVE ARCHIVED RESOLVED	Status temuan saat ini.	String	 Tida	 Ya	 Ya
kesalahan	Menunjukkan kesalahan yang dilaporkan untuk temuan.	String	 Ya	 Ya	 Ya

Kriteria	Deskripsi	Jenis	Aturan arsip	Daftar temuan	Daftar temuan pratinjau akses
Principal .aws	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
principal .federated	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
kondisi.aws: Principal Arn	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

Kriteria	Deskripsi	Jenis	Aturan arsip	Daftar temuan	Daftar temuan pratinjau akses
condition .aws: ID Principal Org	Pengidentifikasi organisasi pengganggu yang jawab ditunjukkan sebagai kondisi untuk akses sumber daya. Untuk mempelajari lebih lanjut, lihat kunci konteks syarat global AWS .	String	 Ya	 Ya	 Ya
kondisi.a ws: Principal 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
kondisi.a ws: SourceIp	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

Kriteria	Deskripsi	Jenis	Aturan arsip	Daftar temuan	Daftar temuan pratinjau akses
kondisi.aws:SourceVpc	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
kondisi.aws:UserId	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
condition.amazonaws.com:identity.cognito-identity	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

Kriteria	Deskripsi	Jenis	Aturan arsip	Daftar temuan	Daftar temuan pratinjau akses
condition.graph.facebook.com:app_id	ID aplikasi Facebook (atau ID situs) yang ditentukan sebagai kondisi untuk memungkinkan Masuk dengan akses federasi Facebook ke peran IAM di temuan. Untuk mempelajari selengkapnya, lihat IAM dan AWS STS kunci-kunci konteks kondisi .	String	 Ya	 Ya	 Ya
condition.accounts.google.com:id	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
kondisi.kms:CallerAccount	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

Kriteria	Deskripsi	Jenis	Aturan arsip	Daftar temuan	Daftar temuan pratinjau akses
kondisi.www.amazon.com:appid	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
id	ID temuan.	String	 Tida	 Ya	 Ya
ChangeType	Menyediakan konteks tentang bagaimana temuan pratinjau akses dibandingkan dengan akses yang ada yang diidentifikasi dalam IAM Access Analyzer.	String	 Tida	 Tida	 Ya
existingFindingId	ID temuan yang ada di IAM Access Analyzer, disediakan hanya untuk temuan yang ada di pratinjau akses.	String	 Tida	 Tida	 Ya
existingFindingStatus	Status yang ada dari temuan, disediakan hanya untuk temuan yang ada di pratinjau akses.	String	 Tida	 Tida	 Ya

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 layanan, 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 tautan untuk melihat dokumentasi peran tertaut 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 `AWSServiceRoleForAccessAnalyzer` 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 lebih lanjut, 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 AWS Management Console 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 API AWS, buat peran yang terhubung dengan layanan dengan nama layanan `access-analyzer.amazonaws.com`. 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 `AWSServiceRoleForAccessAnalyzer` layanan. Setelah Anda membuat peran terkait layanan, Anda tidak dapat mengubah nama peran karena berbagai entitas mungkin mereferensikan peran tersebut. Namun, Anda dapat mengedit penjelasan peran menggunakan IAM. Untuk informasi lebih lanjut, lihat [Mengedit Peran Tertaut Layanan](#) di 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.

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 `AWSServiceRoleForAccessAnalyzer`

1. Buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Di bagian Laporan akses, di bawah Penganalisis akses, pilih Penganalisis.
3. Pilih kotak centang di sebelah kiri atas daftar penganalisis di tabel Penganalisis untuk memilih semua penganalisis.
4. Pilih Hapus.
5. Untuk mengonfirmasi bahwa Anda ingin menghapus penganalisis, masukkan **delete**, lalu pilih Hapus.

Untuk menghapus peran terkait layanan secara manual menggunakan IAM

Gunakan konsol IAM, theAWS CLI, atau AWS API untuk menghapus peran `AWSServiceRoleForAccessAnalyzer` terkait layanan. Untuk informasi lebih lanjut, lihat [Menghapus Peran Tertaut 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 IAM Access Analyzer untuk melihat pratinjau akses publik dan lintas akun untuk bucket Amazon S3, AWS KMS kunci, peran IAM, antrian Amazon SQS, dan rahasia Secrets Manager dengan memberikan izin yang diusulkan untuk sumber daya Anda.

Topik

- [Pratinjau akses di konsol Amazon S3](#)
- [Akses pratinjau dengan IAM Access Analyzer API](#)

Pratinjau akses di konsol Amazon S3

Setelah Anda menyelesaikan kebijakan bucket Anda di konsol Amazon S3, Anda memiliki opsi untuk melihat akses publik dan lintas akun ke bucket Amazon S3. 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 melakukan pratinjau akses. Untuk informasi selengkapnya tentang mengaktifkan IAM Access Analyzer dan izin yang diperlukan, lihat [Mengaktifkan IAM Access Analyzer](#).

Untuk melihat pratinjau akses ke bucket Amazon S3 saat Anda membuat atau mengedit kebijakan bucket

1. Setelah Anda 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 S3, bersama-sama dengan izin bucket yang ada. Ini

- termasuk pengaturan bucket dan akun BPA, bucket ACL, titik akses Amazon S3, dan titik akses multi-wilayah yang melekat pada bucket, serta kebijakan mereka dan pengaturan BPA.
3. Setelah pratinjau akses selesai, pratinjau temuan IAM Access Analyzer ditampilkan. Setiap temuan melaporkan instans penanggung jawab di luar akun dengan akses ke bucket Anda setelah Anda menyimpan kebijakan. 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. Diselesaikan - menunjukkan temuan untuk akses eksternal yang ada yang kebijakan akan dihapus.
 - 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. Sudah Ada — menunjukkan temuan yang ada untuk akses eksternal yang akan tetap tidak berubah.
 - e. Publik — jika temuan adalah untuk akses publik ke sumber daya, temuan ini akan memiliki badge publik, selain salah satu badge 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.

Akses pratinjau dengan IAM Access Analyzer API

Anda dapat menggunakan [IAM Access Analyzer API](#) untuk melihat pratinjau akses publik dan lintas akun untuk bucket Amazon S3, AWS KMS kunci, peran IAM, antrean 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 AccAn dan melakukan pratinjau akses. Untuk informasi selengkapnya tentang mengaktifkan IAM Access Analyzer dan izin yang diperlukan, lihat [Mengaktifkan IAM 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`. `changeType` menyediakan konteks tentang bagaimana temuan pratinjau akses dibandingkan dengan akses yang ada diidentifikasi dalam IAM AccAn:

- Baru - temuan ini untuk akses yang baru diperkenalkan.
- Tidak berubah - temuan pratinjau adalah temuan yang ada yang akan tetap tidak berubah.
- Diubah - 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` adalah `Unchanged` atau `Diubah`, temuan juga akan berisi ID yang ada dan status temuan di IAM AccAn. 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

Untuk membuat pratinjau akses untuk bucket Amazon S3 atau yang sudah ada yang Anda miliki, Anda dapat mengusulkan konfigurasi bucket dengan menentukan kebijakan bucket Amazon S3, ACL bucket, pengaturan BPA bucket, pengaturan BPA bucket, pengaturan BPA bucket, pengaturan BPA bucket, dan titik akses Amazon S3, yang melekat pada bucket.

Note

Sebelum mencoba membuat pratinjau akses untuk bucket baru, kami sarankan Anda memanggil [HeadBucket](#) operasi Amazon S3 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 adalah untuk bucket Amazon S3 yang ada dan Anda tidak menentukan kebijakan bucket Amazon S3, pratinjau akses menggunakan kebijakan yang ada yang melekat pada bucket. Jika pratinjau akses adalah untuk sumber daya baru dan Anda tidak menentukan kebijakan bucket Amazon S3, pratinjau akses 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](#).

Izin akses ACL bucket - Anda dapat mengusulkan hingga 100 izin ACL per bucket. 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 bucket, termasuk hingga sepuluh titik akses baru yang dapat Anda usulkan per bucket. Jika konfigurasi titik akses Amazon S3 yang diusulkan adalah untuk bucket yang ada, pratinjau akses menggunakan konfigurasi titik akses yang diusulkan di tempat 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](#).

Memblokir konfigurasi akses publik — Jika konfigurasi yang diusulkan adalah untuk bucket Amazon S3 yang ada dan Anda tidak menentukan konfigurasi, pratinjau akses menggunakan pengaturan yang ada. Jika konfigurasi yang diusulkan adalah untuk bucket baru dan Anda tidak menentukan konfigurasi BPA bucket, pratinjau akses menggunakan `false`. Jika konfigurasi yang diusulkan adalah untuk titik akses atau multi-wilayah, dan Anda tidak menentukan konfigurasi BPA titik akses, pratinjau akses menggunakan `true`.

Pratinjau akses ke kunci AWS KMS

Untuk membuat pratinjau akses untuk kunci AWS KMS atau kunci AWS KMS yang sudah ada yang Anda miliki, Anda dapat mengusulkan konfigurasi kunci AWS KMS dengan menentukan kebijakan kunci dan konfigurasi izin AWS KMS.

AWS KMSkebijakan kunci - Jika konfigurasi untuk bukti kunci yang ada dan Anda tidak menentukan kebijakan kunci, pratinjau akses menggunakan kebijakan yang ada untuk kunci. 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 KMSizin - Analisis mendukung izin hingga 100 KMS per konfigurasi*. * Jika konfigurasi izin diusulkan adalah untuk kunci yang ada, pratinjau akses menggunakan daftar konfigurasi izin yang diusulkan di tempat izin yang ada. Jika tidak, pratinjau akses menggunakan izin yang ada untuk kunci.

Pratinjau akses ke peran IAM

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 adalah untuk peran IAM baru, Anda harus menentukan kebijakan kepercayaan. Jika konfigurasi adalah untuk peran IAM yang ada yang Anda miliki dan Anda tidak mengusulkan kebijakan kepercayaan, pratinjau akses menggunakan kebijakan kepercayaan yang ada untuk peran. Kebijakan kepercayaan yang diusulkan tidak boleh string kosong.

Pratinjau akses ke antrean Amazon SQS

Untuk membuat pratinjau akses untuk antrean Amazon SQS atau antrean Amazon SQS yang sudah ada yang Anda miliki, Anda dapat mengusulkan konfigurasi antrean Amazon SQS dengan menentukan kebijakan Amazon SQS untuk antrean.

Kebijakan antrean Amazon SQS — Jika konfigurasi adalah 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 adalah untuk sumber daya baru dan Anda tidak menentukan kebijakan, pratinjau akses mengasumsikan antrean Amazon SQS tanpa kebijakan. Untuk mengusulkan penghapusan kebijakan antrean 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. 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 AWS KMS kunci ID kunci, pratinjau akses menggunakan kunci KMS AWS default. Jika Anda menentukan string kosong untuk ID AWS KMS kunci, pratinjau akses menggunakan kunci KMS default dari AWS akun.

Cek 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 AWS praktik 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 [IAM Access Analyzer](#).

Topik

- [Validasi kebijakan IAM Access Analyzer](#)
- [Pemeriksaan kebijakan khusus IAM Access Analyzer](#)

Validasi kebijakan 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 kebijakan Access Analyzer](#)


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 AWS Management Console 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 [Menyunting kebijakan terkelola pelanggan \(konsol\)](#).
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. Opsional. 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 [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. centang kotak 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 kebijakan 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. 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, lihat [Validasi kebijakan IAM Access Analyzer](#)

Kesalahan - Akun ARN tidak diizinkan

Dalam AWS Management Console, 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 atau, 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 menentukan 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 akun. AWS 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

Dalam AWS Management Console, 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 atau, 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 dikosongkan. 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

Dalam AWS Management Console, 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 atau, 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 IAM JSON: Operator kondisi](#)

Kesalahan - Kunci duplikat dengan kasus yang berbeda

Dalam AWS Management Console, 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 atau, 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 kasus nilai kunci kondisi tergantung pada operator kondisi yang Anda gunakan. Untuk informasi selengkapnya tentang sensitivitas huruf besar/kecil pada tombol kondisi, lihat. [Elemen kebijakan IAM JSON: Condition](#)

Istilah terkait

- [Ketentuan](#)
- [Blok kondisi](#)
- [Kunci kondisi global](#)
- [AWS kunci kondisi layanan](#)

Kesalahan - Tindakan tidak valid

Dalam AWS Management Console, 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 atau, 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. Ini bisa 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 terkelola dengan kesalahan ini

[AWS kebijakan terkelola](#) memungkinkan Anda memulai AWS dengan menetapkan izin berdasarkan kasus AWS penggunaan umum.

Kebijakan AWS terkelola berikut mencakup 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, AWS sarankan agar Anda menghapus tindakan yang tidak valid dari kebijakan Anda.

- [FullAccessPolicyAmazonEMR_v2](#)
- [CloudWatchSyntheticsFullAccess](#)

Kesalahan - Akun ARN tidak valid

Dalam AWS Management Console, 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 atau, 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, lihat [Menemukan ID AWS akun](#).

Istilah terkait

- [Sumber daya kebijakan](#)
- [Pengidentifikasi Akun](#)
- [ARN Sumber Daya](#)
- [AWS sumber daya layanan dengan format ARN](#)

Kesalahan — Awalan ARN tidak valid

Dalam AWS Management Console, 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 atau, 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

Dalam AWS Management Console, 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 atau, 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

Dalam AWS Management Console, 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 atau, 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 [Tindakan, 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

Dalam AWS Management Console, 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 atau, 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 di sumber daya ARN harus sesuai dengan spesifikasi (termasuk kapitalisasi) untuk awalan layanan. Untuk melihat awalan untuk layanan, lihat [Tindakan, sumber daya, dan kunci kondisi untuk AWS layanan](#). Pilih nama layanan dan temukan awalannya di kalimat pertama.

Istilah terkait

- [Sumber daya kebijakan](#)
- [ARN Sumber Daya](#)
- [AWS sumber daya layanan dengan format ARN](#)

Kesalahan - Tipe data kondisi tidak valid

Dalam AWS Management Console, 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 atau, 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 dari 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 tombol kondisi untuk layanan tersebut.

Misalnya, kunci kondisi [CurrentTime](#) global mendukung operator Date kondisi. Jika Anda memberikan string atau bilangan bulat untuk nilai di blok kondisi, tipe data tidak akan cocok.

Istilah terkait

- [Ketentuan](#)
- [Blok kondisi](#)
- [Elemen kebijakan IAM JSON: Operator kondisi](#)
- [Kunci kondisi global](#)
- [AWS kunci kondisi layanan](#)

Kesalahan - Format kunci kondisi tidak valid

Dalam AWS Management Console, 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 atau, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The condition key format is not valid. Use the format service:keyname."
```

Menyelesaikan kesalahan

Kunci dalam kondisi pasangan kunci-nilai 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 tombol kondisi untuk layanan tersebut.

Istilah terkait

- [Ketentuan](#)
- [Kunci kondisi global](#)
- [AWS kunci kondisi layanan](#)

Kesalahan - Kondisi tidak valid beberapa Boolean

Dalam AWS Management Console, 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 atau, 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. Saat 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 tombol kondisi untuk layanan tersebut.

- [Ketentuan](#)

- [Kunci kondisi global](#)
- [AWS kunci kondisi layanan](#)

Kesalahan - Operator kondisi tidak valid

Dalam AWS Management Console, 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 atau, 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 IAM JSON: Operator kondisi](#)
- [Elemen kondisi](#)
- [Ikhtisar kebijakan JSON](#)

Kesalahan - Efek tidak valid

Dalam AWS Management Console, 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 atau, 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

Dalam AWS Management Console, 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 atau, 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 pada 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

Dalam AWS Management Console, 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 atau, 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 ARN sumber daya 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

Dalam AWS Management Console, 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 atau, 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

Dalam AWS Management Console, 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 atau, 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 prinsipal untuk menggunakan format pasangan nilai kunci yang didukung.

Anda dapat menentukan prinsipal dalam kebijakan berbasis sumber daya, tetapi bukan kebijakan berbasis identitas.

Misalnya, untuk menentukan akses bagi semua orang di AWS akun, gunakan prinsip berikut dalam kebijakan Anda:

```
"Principal": { "AWS": "123456789012" }
```

Istilah terkait

- [Elemen kebijakan JSON: Principal](#)
- [Kebijakan berbasis identitas dan kebijakan berbasis sumber daya](#)

Kesalahan - Kunci utama tidak valid

Dalam AWS Management Console, 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 atau, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The principal key {{principal-key}} is not valid."
```

Menyelesaikan kesalahan

Perbarui kunci dalam pasangan nilai kunci utama untuk menggunakan kunci utama yang didukung. Berikut ini adalah kunci utama yang didukung:

- AWS
- CanonicalUser
- Federasi
- Layanan

Istilah terkait

- [Elemen utama](#)

Kesalahan - Wilayah Tidak Valid

Dalam AWS Management Console, 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 atau, 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

Dalam AWS Management Console, 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 atau, 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 untuk layanan, lihat [Tindakan, sumber daya, dan kunci kondisi untuk AWS layanan](#). Pilih nama layanan dan temukan awalnya di kalimat pertama.

Istilah terkait

- [Layanan yang dikenal dan tindakan, sumber daya, dan kunci kondisinya](#)

Kesalahan - Kunci kondisi layanan tidak valid

Dalam AWS Management Console, 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 atau, 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 untuk layanan, lihat [Tindakan, 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

Dalam AWS Management Console, 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 atau, 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 untuk layanan, lihat [Tindakan, sumber daya, dan kunci kondisi untuk AWS layanan](#). Pilih nama layanan dan temukan awalannya di kalimat pertama.

Istilah terkait

- [Elemen aksi](#)
- [Layanan yang dikenal dan tindakan mereka](#)

Kesalahan - Variabel tidak valid untuk operator

Dalam AWS Management Console, 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 atau, 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 ketika 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 Numerik, Tanggal, Boolean, Biner, Alamat IP, atau operator Null.

Istilah terkait

- [Menggunakan variabel kebijakan dalam elemen Kondisi](#)
- [Elemen kondisi](#)

Kesalahan - Versi tidak valid

Dalam AWS Management Console, 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 atau, 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 di semua kebijakan Anda.

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

Istilah terkait

- [Elemen versi](#)

Kesalahan - Kesalahan sintaks Json

Dalam AWS Management Console, 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 atau, 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

- [JSON validator](#)
- [Referensi elemen kebijakan IAM JSON](#)
- [Ikhtisar kebijakan JSON](#)

Kesalahan - Kesalahan sintaks Json

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Json syntax error: Fix the JSON syntax error.
```

Dalam panggilan terprogram ke AWS API AWS CLI atau, 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

- [JSON validator](#)
- [Referensi elemen kebijakan IAM JSON](#)
- [Ikhtisar kebijakan JSON](#)

Kesalahan - Tindakan hilang

Dalam AWS Management Console, 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 atau, 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 aksi](#)
- [NotAction elemen](#)
- [Ikhtisar kebijakan JSON](#)

Kesalahan - Bidang ARN tidak ada

Dalam AWS Management Console, 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 atau, 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 di ARN sumber daya harus sesuai dengan spesifikasi untuk jenis sumber daya yang diketahui. Untuk melihat format ARN yang diharapkan untuk layanan, lihat [Tindakan, 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

Dalam AWS Management Console, 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 atau, 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

Dalam AWS Management Console, 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 atau, 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 menyertakan Effect elemen dengan nilai **Allow** dan **Deny**.

Istilah terkait

- [Elemen efek](#)
- [Ikhtisar kebijakan JSON](#)

Kesalahan - Kepala sekolah tidak ada

Dalam AWS Management Console, 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 atau, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Add a Principal element to the policy statement."
```

Menyelesaikan kesalahan

Kebijakan berbasis sumber daya harus menyertakan elemen. **Principal**

Misalnya, untuk menentukan akses bagi semua orang di AWS akun, gunakan prinsip berikut dalam kebijakan Anda:

```
"Principal": { "AWS": "123456789012" }
```

Istilah terkait

- [Elemen utama](#)
- [Kebijakan berbasis identitas dan kebijakan berbasis sumber daya](#)

Kesalahan - Kualifikasi tidak ada

Dalam AWS Management Console, 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 atau, 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 satu kunci kondisi, Anda harus menyertakan kondisi dalam tanda kurung seperti array ("Key2": ["Value2A", "Value2B"]). Anda juga harus menggunakan ForAllValues atau ForAnyValue mengatur operator 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 multivaluasi](#)
- [Elemen kondisi](#)

AWS kebijakan terkelola dengan kesalahan ini

[AWS kebijakan terkelola](#) memungkinkan Anda memulai AWS dengan menetapkan izin berdasarkan kasus AWS penggunaan umum.

Kebijakan AWS terkelola berikut menyertakan kualifikasi yang hilang untuk kunci kondisi dalam pernyataan kebijakan mereka. Saat menggunakan kebijakan AWS terkelola sebagai referensi untuk membuat kebijakan terkelola pelanggan, AWS sarankan Anda menambahkan `ForAllValues` atau `ForAnyValue` mengkondisikan kualifikasi kunci ke `Condition` elemen Anda.

- [AWSGlueConsoleSageMakerNotebookFullAccess](#)

Kesalahan - Sumber daya hilang

Dalam AWS Management Console, 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 atau, 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](#)
- [Kebijakan berbasis identitas dan kebijakan berbasis sumber daya](#)
- [Ikhtisar kebijakan JSON](#)

Kesalahan - Pernyataan hilang

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Missing statement: Add a statement to the policy
```

Dalam panggilan terprogram ke AWS API AWS CLI atau, 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

Dalam AWS Management Console, 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 atau, 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 hadir 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 aksi kesalahan sintaks SCP

Dalam AWS Management Console, 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 atau, 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 Action atau NotAction elemen. Namun, nilai-nilai ini dapat mencakup wildcard (*) hanya di akhir string. Ini berarti Anda dapat menentukan `iam:Get*` tetapi tidak `iam:*role`.

Untuk menentukan beberapa tindakan, AWS sarankan Anda mencantumkanannya satu per satu.

Istilah terkait

- [SCP Aksi dan elemen NotAction](#)
- [Evaluasi SCP](#)
- [AWS Organizations kebijakan kontrol layanan](#)
- [Elemen kebijakan IAM JSON: Tindakan](#)

Kesalahan - Kesalahan sintaks SCP memungkinkan kondisi

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
SCP syntax error allow condition: SCPs do not support the Condition element with effect Allow. Update the element Condition or the effect.
```

Dalam panggilan terprogram ke AWS API AWS CLI atau, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "SCPs do not support the Condition element with effect Allow. Update the element Condition or the effect."
```

Menyelesaikan kesalahan

AWS Organizations kebijakan kontrol layanan (SCP) mendukung menentukan nilai dalam Condition elemen hanya ketika Anda menggunakan. "Effect": "Deny"

Untuk mengizinkan hanya satu tindakan, Anda dapat menolak akses ke semuanya kecuali kondisi yang Anda tentukan menggunakan `...NotEquals` versi operator kondisi. Ini meniadakan perbandingan yang dibuat oleh operator.

Istilah terkait

- [Elemen Kondisi SCP](#)
- [Evaluasi SCP](#)
- [AWS Organizations kebijakan kontrol layanan](#)
- [Contoh kebijakan: Menolak akses AWS berdasarkan Wilayah yang diminta](#)
- [Elemen kebijakan IAM JSON: Operator kondisi](#)
- [Elemen kebijakan IAM JSON: Kondisi](#)

Kesalahan - Kesalahan sintaks SCP memungkinkan NotAction

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
SCP syntax error allow NotAction: SCPs do not support NotAction with effect Allow. Update the element NotAction or the effect.
```

Dalam panggilan terprogram ke AWS API AWS CLI atau, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "SCPs do not support NotAction with effect Allow. Update the element NotAction or the effect."
```

Menyelesaikan kesalahan

AWS Organizations kebijakan kontrol layanan (SCP) tidak mendukung penggunaan `NotAction` elemen dengan `"Effect": "Allow"`.

Anda harus menulis ulang logika untuk mengizinkan daftar tindakan, atau untuk menolak setiap tindakan yang tidak terdaftar.

Istilah terkait

- [SCP Aksi dan elemen NotAction](#)
- [Evaluasi SCP](#)
- [AWS Organizations kebijakan kontrol layanan](#)
- [Elemen kebijakan IAM JSON: Tindakan](#)

Kesalahan - Kesalahan sintaks SCP memungkinkan sumber daya

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
SCP syntax error allow resource: SCPs do not support Resource with effect Allow. Update the element Resource or the effect.
```

Dalam panggilan terprogram ke AWS API AWS CLI atau, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "SCPs do not support Resource with effect Allow. Update the element Resource or the effect."
```

Menyelesaikan kesalahan

AWS Organizations kebijakan kontrol layanan (SCP) mendukung menentukan nilai dalam `Resource` elemen hanya ketika Anda menggunakan `"Effect": "Deny"`

Anda harus menulis ulang logika untuk memungkinkan semua sumber daya, atau untuk menolak setiap sumber daya yang terdaftar.

Istilah terkait

- [Elemen Sumber Daya SCP](#)
- [Evaluasi SCP](#)
- [AWS Organizations kebijakan kontrol layanan](#)

- [Elemen kebijakan IAM JSON: Sumber daya](#)

Kesalahan - Kesalahan sintaks SCP NotResource

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
SCP syntax error NotResource: SCPs do not support the NotResource element. Update the policy to use Resource instead.
```

Dalam panggilan terprogram ke AWS API AWS CLI atau, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "SCPs do not support the NotResource element. Update the policy to use Resource instead."
```

Menyelesaikan kesalahan

AWS Organizations kebijakan kontrol layanan (SCP) tidak mendukung NotResource elemen.

Anda harus menulis ulang logika untuk memungkinkan semua sumber daya, atau untuk menolak setiap sumber daya yang terdaftar.

Istilah terkait

- [Elemen Sumber Daya SCP](#)
- [Evaluasi SCP](#)
- [AWS Organizations kebijakan kontrol layanan](#)
- [Elemen kebijakan IAM JSON: Sumber daya](#)

Kesalahan - Prinsip kesalahan sintaks SCP

Dalam AWS Management Console, 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 atau, 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 `Principal` atau `NotPrincipal` elemen.

Anda dapat menentukan Amazon Resource Name (ARN) menggunakan kunci kondisi `aws:PrincipalArn` global dalam elemen. `Condition`

Istilah terkait

- [Sintaks SCP](#)
- [Kunci kondisi global untuk kepala sekolah](#)

Kesalahan - Sids Unik diperlukan

Dalam AWS Management Console, 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 atau, 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` (ID pernyataan) memungkinkan Anda memasukkan pengenal 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, `Sid` nilainya hanyalah sub-ID dari ID dokumen kebijakan. Misalnya, di IAM, `Sid` nilainya harus unik dalam kebijakan JSON.

Istilah terkait

- [Elemen kebijakan IAM JSON: Sid](#)

Kesalahan — Tindakan yang tidak didukung dalam kebijakan

Dalam AWS Management Console, 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 atau, 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 pada kebijakan berbasis sumber daya Anda.

Istilah terkait

- [Elemen kebijakan JSON: Tindakan](#)

Kesalahan - Kombinasi elemen yang tidak didukung

Dalam AWS Management Console, 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 atau, 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-sama. 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 IAM JSON](#)
- [Ikhtisar kebijakan JSON](#)

Kesalahan - Kunci kondisi global yang tidak didukung

Dalam AWS Management Console, 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 atau, 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 `aws:PrincipalArn` atau kunci kondisi `aws:SourceArn` global. Misalnya `aws:ARN`, gunakan `aws:PrincipalArn` untuk membandingkan Amazon Resource Name (ARN) 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) sumber daya yang membuat service-to-service permintaan dengan ARN yang Anda tentukan dalam kebijakan.

Istilah terkait

- [AWS kunci konteks kondisi global](#)

Kesalahan — Prinsipal yang tidak didukung

Dalam AWS Management Console, 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 atau, 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

`PrincipalElement` menentukan prinsipal 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. Kebijakan berbasis sumber daya adalah kebijakan yang diterapkan langsung ke sumber daya. Misalnya, Anda dapat menyematkan kebijakan di bucket Amazon S3 atau AWS kunci KMS.

Istilah terkait

- [AWS Elemen kebijakan JSON: Principal](#)
- [Akses sumber daya lintas akun di IAM](#)

Kesalahan — ARN sumber daya yang tidak didukung dalam kebijakan

Dalam AWS Management Console, 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 atau, 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 pada kebijakan berbasis sumber daya Anda.

Istilah terkait

- [Elemen kebijakan JSON: Tindakan](#)

Kesalahan — Sid Tidak Didukung

Dalam AWS Management Console, 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 atau, 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 IAM JSON: Sid](#)

Kesalahan - Wildcard tidak didukung pada prinsipnya

Dalam AWS Management Console, 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 atau, 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 penggunaan pasangan kunci-nilai. Nilai pokok yang ditentukan dalam polis termasuk wildcard (*). Anda tidak dapat menyertakan wildcard dengan kunci utama yang Anda tentukan. Misalnya, saat Anda menentukan pengguna dalam `Principal` elemen, Anda tidak dapat menggunakan wildcard yang 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 yang berarti “semua sesi”. Anda harus menyebutkan sesi tertentu. Anda juga tidak dapat menggunakan wildcard untuk mencocokkan bagian dari nama atau ARN.

Untuk mengatasi temuan ini, hapus wildcard dan berikan prinsipal yang lebih spesifik.

Istilah terkait

- [AWS Elemen kebijakan JSON: Principal](#)

Kesalahan - Penjepit tidak ada dalam variabel

Dalam AWS Management Console, 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 atau, 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 kurawal kurawal (). { } Di dalam \${ } karakter, sertakan nama nilai dari permintaan yang ingin Anda gunakan dalam kebijakan.

Untuk mengatasi temuan ini, tambahkan penjepit yang hilang untuk memastikan set kawat gigi pembukaan dan penutup penuh hadir.

Istilah terkait

- [Elemen kebijakan IAM: Variabel](#)

Kesalahan - Kutipan hilang dalam variabel

Dalam AWS Management Console, 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 atau, 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 prinsipal diberi tagteam=yellow, mereka dapat mengakses bucket *DOC-EXAMPLE-BUCKET* Amazon S3 dengan nama tersebut. *DOC-EXAMPLE-BUCKET*-yellow Kebijakan dengan sumber daya ini mungkin memungkinkan anggota tim untuk mengakses sumber daya mereka sendiri, tetapi bukan milik tim lain. Untuk pengguna tanpa tag tim, Anda dapat menetapkan nilai defaultcompany-wide. Pengguna ini hanya dapat mengakses *DOC-EXAMPLE-BUCKET*-company-wide bucket di mana mereka dapat melihat informasi yang luas, seperti instruksi untuk bergabung dengan tim.

```
"Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET-${aws:PrincipalTag/team, 'company-wide'}"
```

Istilah terkait

- [Elemen kebijakan IAM: Variabel](#)

Kesalahan - Ruang yang tidak didukung dalam variabel

Dalam AWS Management Console, 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 atau, 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 kurawal kurawal (). { } 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

Dalam AWS Management Console, 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 atau, 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 kurawal kurawal (). { } 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

Dalam AWS Management Console, 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 atau, 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

Dalam AWS Management Console, 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 atau, 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 menyetelnya 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 IAM JSON: Versi](#)

Kesalahan - Alamat IP pribadi

Dalam AWS Management Console, 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 atau, 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 ketika kebijakan Anda hanya mengizinkan alamat IP pribadi. Dalam hal ini, kondisinya tidak akan pernah cocok.

- [aws: kunci kondisi SourceIp global](#)
- [Elemen kebijakan IAM JSON: Kondisi](#)

Kesalahan - Pribadi NotIpAddress

Dalam AWS Management Console, 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 atau, 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 IAM JSON: Kondisi](#)

Kesalahan — Ukuran kebijakan melebihi kuota SCP

Dalam AWS Management Console, 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 atau, 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 `Action` atau `NotAction` elemen. Namun, nilai-nilai ini dapat mencakup wildcard (*) hanya di akhir string. Ini berarti Anda dapat menentukan `iam:Get*` tetapi tidak `iam:*role`.

Untuk menentukan beberapa tindakan, AWS sarankan Anda mencantulkannya satu per satu.

Istilah terkait

- [Kuota untuk Organizations AWS](#)
- [AWS Organizations kebijakan kontrol layanan](#)

Kesalahan - Format utama layanan tidak valid

Dalam AWS Management Console, 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 atau, 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 cocok dengan format utama 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 masing-masing layanan.

Pengenalan untuk kepala 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

- [Prinsipal layanan](#)
- [AWS kunci kondisi global](#)
- [kms:ViaServicekunci kondisi](#)

Kesalahan - Kunci tag hilang dalam kondisi

Dalam AWS Management Console, 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 atau, 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](#)
- [Ketentuan](#)
- [Kunci kondisi global](#)
- [AWS kunci kondisi layanan](#)

Kesalahan - Format vpc tidak valid

Dalam AWS Management Console, 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 atau, 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

Kunci `aws:SourceVpc` kondisi harus menggunakan awalan `vpc-` diikuti oleh 8 atau 17 karakter alfanumerik, misalnya, atau. `vpc-11223344556677889` `vpc-12345678`

Istilah terkait

- [AWS kunci kondisi global: aws: SourceVpc](#)

Kesalahan - Format vpce tidak valid

Dalam AWS Management Console, 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 atau, 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

Kunci `aws:SourceVpce` kondisi harus menggunakan awalan `vpce-` diikuti oleh 8 atau 17 karakter alfanumerik, misalnya, `vpce-11223344556677889` `vpce-12345678`

Istilah terkait

- [AWS kunci kondisi global: `aws:SourceVpce`](#)

Kesalahan - Prinsipal federasi tidak didukung

Dalam AWS Management Console, 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 atau, 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

`PrincipalElement` ini menggunakan prinsip-prinsip federasi untuk kebijakan kepercayaan yang melekat pada peran IAM untuk menyediakan akses melalui federasi identitas. Kebijakan identitas dan kebijakan berbasis sumber daya lainnya tidak mendukung penyedia identitas federasi di elemen tersebut. `Principal` Misalnya, Anda tidak dapat menggunakan prinsipal SAFL 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 yang tidak didukung untuk kunci kondisi

Dalam AWS Management Console, 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 atau, 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 diizinkan atau ditolak secara efektif 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

Dalam AWS Management Console, 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 atau, 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 pada kebijakan berbasis sumber daya Anda.

Istilah terkait

- [Elemen kebijakan JSON: Tindakan](#)

Kesalahan — ARN sumber daya yang tidak didukung dalam kebijakan

Dalam AWS Management Console, 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 atau, 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 pada kebijakan berbasis sumber daya Anda.

Istilah terkait

- [Elemen kebijakan JSON: Tindakan](#)

Kesalahan - Kunci kondisi yang tidak didukung untuk prinsipal layanan

Dalam AWS Management Console, 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 atau, 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 pengenal untuk layanan. Anda tidak dapat menggunakan beberapa kunci kondisi dengan prinsip layanan tertentu. Misalnya, Anda tidak dapat menggunakan kunci `aws:PrincipalOrgID` kondisi dengan kepala `layanancloudfront.amazonaws.com`. Anda harus menghapus kunci kondisi yang tidak berlaku untuk prinsip layanan dalam `Principal` elemen.

Istilah terkait

- [Prinsipal layanan](#)
- [Elemen kebijakan JSON: Principal](#)

Kesalahan — Kesalahan sintaks kebijakan kepercayaan peran `notprincipal`

Dalam AWS Management Console, 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 atau, 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 pada prinsipnya

Dalam AWS Management Console, 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 atau, 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 kesalahan sintaks kebijakan kepercayaan peran

Dalam AWS Management Console, 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 atau, 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 melekat padanya. 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 - Ketik rentang IP ketidakcocokan

Dalam AWS Management Console, 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 atau, 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 IAM JSON: Operator kondisi](#)

Kesalahan - Tindakan tidak ada untuk kunci kondisi

Dalam AWS Management Console, 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 atau, 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 diizinkan atau ditolak secara efektif oleh kebijakan Anda, tambahkan tindakan ke `Action` elemen.

Istilah terkait

- [Elemen kebijakan JSON: Tindakan](#)

Kesalahan - Sintaks utama federasi tidak valid dalam kebijakan kepercayaan peran

Dalam AWS Management Console, 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 atau, 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 pokok menentukan prinsip federasi yang tidak sesuai dengan format yang diharapkan. Perbarui format prinsipal federasi ke nama domain yang valid atau metadata SAMP ARN.

Istilah terkait

- [Pengguna dan peran federasi](#)

Kesalahan - Tindakan tidak cocok untuk prinsipal

Dalam AWS Management Console, 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 atau, 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 prinsipal yang ditentukan dalam elemen Principal. Misalnya, Anda tidak dapat menggunakan prinsipal penyedia SAMP dengan `sts:AssumeRoleWithWebIdentity` tindakan tersebut. Anda harus menggunakan prinsipal penyedia SAMP dengan `sts:AssumeRoleWithSAML` tindakan atau menggunakan prinsipal penyedia OIDC dengan tindakan tersebut.

`sts:AssumeRoleWithWebIdentity`

Istilah terkait

- [AssumeRoleWithSAM](#)
- [AssumeRoleWithWebIdentity](#)

Kesalahan - Tindakan tidak ada untuk peran di mana saja kebijakan kepercayaan

Dalam AWS Management Console, 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 atau, 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 AWS kredensial sementara, peran tersebut harus mempercayai kepala layanan IAM Roles Anywhere. Prinsipal 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 AWS Identity and Access Management Peran Di Mana Saja](#)

Peringatan Umum - Buat SLR dengan NotResource

Dalam AWS Management Console, 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 atau, 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

Tindakan tersebut `iam:CreateServiceLinkedRole` memberikan izin untuk membuat peran IAM yang memungkinkan AWS layanan melakukan tindakan atas nama Anda. Menggunakan `iam:CreateServiceLinkedRole` 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 dalam `Resource` elemen sebagai gantinya.

- [CreateServiceLinkedRole operasi](#)
- [Elemen kebijakan IAM JSON: NotResource](#)
- [Elemen kebijakan IAM JSON: Sumber daya](#)

Peringatan Umum - Buat SLR dengan bintang beraksi dan NotResource

Dalam AWS Management Console, 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 atau, 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

Tindakan tersebut `iam:CreateServiceLinkedRole` memberikan izin untuk membuat peran IAM yang memungkinkan AWS layanan melakukan tindakan atas nama Anda. Kebijakan dengan wildcard (*) dalam 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 dalam Resource elemen sebagai gantinya.

- [CreateServiceLinkedRole operasi](#)
- [Elemen kebijakan IAM JSON: NotResource](#)
- [Elemen kebijakan IAM JSON: Sumber daya](#)

Peringatan Umum - Buat SLR dengan NotAction dan NotResource

Dalam AWS Management Console, 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 atau, 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

Tindakan tersebut `iam:CreateServiceLinkedRole` 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 merekomendasikan agar Anda menulis ulang kebijakan untuk mengizinkan `iam:CreateServiceLinkedRole` pada daftar terbatas ARN dalam Resource elemen sebagai gantinya. Anda juga dapat `iam:CreateServiceLinkedRole` menambahkan `NotAction` elemen.

- [CreateServiceLinkedRole operasi](#)
- [Elemen kebijakan IAM JSON: NotAction](#)
- [Elemen kebijakan IAM JSON: Tindakan](#)
- [Elemen kebijakan IAM JSON: NotResource](#)
- [Elemen kebijakan IAM JSON: Sumber daya](#)

Peringatan Umum - Buat SLR dengan bintang di sumber daya

Dalam AWS Management Console, 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 atau, 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

Tindakan tersebut `iam:CreateServiceLinkedRole` memberikan izin untuk membuat peran IAM yang memungkinkan AWS layanan melakukan tindakan atas nama Anda. Menggunakan `iam:CreateServiceLinkedRole` dalam kebijakan dengan wildcard (*) dalam Resource elemen dapat memungkinkan pembuatan peran terkait layanan yang tidak diinginkan untuk beberapa sumber daya. AWS merekomendasikan agar Anda menentukan ARN yang diizinkan dalam Resource elemen sebagai gantinya.

- [CreateServiceLinkedRole operasi](#)
- [Elemen kebijakan IAM JSON: Sumber daya](#)

AWS kebijakan terkelola dengan peringatan umum ini

[AWS kebijakan terkelola](#) memungkinkan Anda memulai AWS dengan menetapkan izin berdasarkan kasus AWS penggunaan umum.

Beberapa kasus penggunaan tersebut adalah untuk pengguna yang kuat di dalam akun Anda. Kebijakan AWS terkelola berikut menyediakan akses pengguna yang kuat dan memberikan izin untuk membuat [peran terkait layanan](#) untuk layanan apa pun. AWS merekomendasikan agar Anda melampirkan kebijakan AWS terkelola berikut hanya ke identitas IAM yang Anda anggap pengguna daya.

- [PowerUserAccess](#)
- [AlexaForBusinessFullAccess](#)
- [AWSOrganizationsServiceTrustPolicy](#)— Kebijakan AWS terkelola ini memberikan izin untuk digunakan oleh peran AWS Organizations terkait layanan. Peran ini memungkinkan Organizations untuk membuat peran terkait layanan tambahan untuk layanan lain di organisasi Anda AWS .

Peringatan Umum - Buat SLR dengan bintang dalam aksi dan sumber daya

Dalam AWS Management Console, 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 atau, 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

Tindakan tersebut `iam:CreateServiceLinkedRole` 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 dalam Resource elemen sebagai gantinya.

- [CreateServiceLinkedRole operasi](#)
- [Elemen kebijakan IAM JSON: Tindakan](#)
- [Elemen kebijakan IAM JSON: Sumber daya](#)

AWS kebijakan terkelola dengan peringatan umum ini

[AWS kebijakan terkelola](#) memungkinkan Anda memulai AWS dengan menetapkan izin berdasarkan kasus AWS penggunaan umum.

Beberapa kasus penggunaan tersebut adalah untuk administrator dalam akun Anda. Kebijakan AWS terkelola berikut menyediakan akses administrator dan memberikan izin untuk membuat [peran terkait layanan](#) untuk layanan apa pun. AWS merekomendasikan agar Anda melampirkan kebijakan AWS terkelola berikut hanya pada identitas IAM yang Anda anggap administrator.

- [AdministratorAccess](#)
- [IAM FullAccess](#)

Peringatan Umum - Buat SLR dengan bintang di sumber daya dan NotAction

Dalam AWS Management Console, 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 atau, 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

Tindakan tersebut `iam:CreateServiceLinkedRole` memberikan izin untuk membuat peran IAM yang memungkinkan AWS layanan melakukan tindakan atas nama Anda. Menggunakan `NotAction` elemen dalam kebijakan dengan wildcard (*) dalam Resource elemen dapat memungkinkan pembuatan peran terkait layanan yang tidak diinginkan untuk beberapa sumber daya. AWS merekomendasikan agar Anda menentukan ARN yang diizinkan dalam Resource elemen sebagai gantinya. Anda juga dapat `iam:CreateServiceLinkedRole` menambahkan `NotAction` elemen.

- [CreateServiceLinkedRole operasi](#)
- [Elemen kebijakan IAM JSON: NotAction](#)
- [Elemen kebijakan IAM JSON: Tindakan](#)
- [Elemen kebijakan IAM JSON: Sumber daya](#)

Peringatan Umum - Kunci kondisi global yang tidak digunakan lagi

Dalam AWS Management Console, 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 atau, 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 mencakup kunci kondisi global yang tidak digunakan lagi. Perbarui kunci kondisi pada pasangan kunci-nilai kondisi untuk menggunakan kunci kondisi global yang didukung.

- [Kunci kondisi global](#)

Peringatan Umum - Nilai tanggal tidak valid

Dalam AWS Management Console, 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 atau, 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 menentukan 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 (1997-07-16T 19:20:30 + 01:00).

- [Format Tanggal dan Waktu W3C](#)
- [Elemen kebijakan IAM JSON: Versi](#)
- [aws: kunci kondisi CurrentTime global](#)

Peringatan Umum - Referensi peran tidak valid

Dalam AWS Management Console, 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 atau, 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 merekomendasikan agar Anda menentukan Nama Sumber Daya Amazon (ARN) untuk peran IAM, bukan 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, itu akan memiliki ID baru, dan kebijakan tidak akan cocok dengan ID peran baru.

- [Menentukan kepala sekolah: peran IAM](#)
- [ARN IAM](#)
- [ID unik IAM](#)

Peringatan Umum - Referensi pengguna tidak valid

Dalam AWS Management Console, 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 atau, 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 merekomendasikan agar Anda menentukan Nama Sumber Daya Amazon (ARN) untuk pengguna IAM, bukan 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, itu akan memiliki ID baru, dan kebijakan tidak akan cocok dengan ID pengguna baru.

- [Menentukan prinsipal: pengguna IAM](#)
- [ARN IAM](#)
- [ID unik IAM](#)

Peringatan Umum - Versi tidak ada

Dalam AWS Management Console, 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 atau, 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 merekomendasikan agar Anda menyertakan `Version` parameter opsional dalam kebijakan Anda. Jika Anda tidak menyertakan elemen `Versi`, nilai default, tetapi fitur yang lebih baru 2012-10-17, 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: Versi](#)

Peringatan Umum - Sids Unik direkomendasikan

Dalam AWS Management Console, 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 atau, 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` (ID pernyataan) memungkinkan Anda memasukkan pengenal 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 IAM JSON: Sid](#)

Peringatan Umum - Wildcard tanpa operator seperti

Dalam AWS Management Console, 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 atau, 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. Bila 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 IAM JSON: Operator kondisi](#)
- [Elemen kebijakan IAM JSON: Kondisi](#)

AWS kebijakan terkelola dengan peringatan umum ini

[AWS kebijakan terkelola](#) memungkinkan Anda memulai AWS dengan menetapkan izin berdasarkan kasus AWS penggunaan umum.

Kebijakan AWS terkelola berikut menyertakan wildcard dalam nilai kondisinya tanpa operator kondisi yang menyertakan `Like` pencocokan pola. Saat menggunakan kebijakan AWS terkelola sebagai referensi untuk membuat kebijakan terkelola pelanggan, AWS sarankan agar Anda menggunakan operator kondisi yang mendukung pencocokan pola dengan wildcard (*,?) , seperti `StringLike`.

- [AWSGlueConsoleSageMakerNotebookFullAccess](#)

Peringatan Umum — Ukuran kebijakan melebihi kuota kebijakan identitas

Dalam AWS Management Console, 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 atau, 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 10 kebijakan terkelola ke identitas IAM (pengguna, grup pengguna, atau peran). Namun, ukuran setiap kebijakan terkelola tidak dapat melebihi kuota default 6.144 karakter. IAM tidak menghitung spasi kosong saat menghitung ukuran kebijakan terhadap kuota ini. Kuota, juga disebut sebagai batas dalam 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 inline per identitas tidak dapat melebihi kuota yang ditentukan.

Jika kebijakan Anda lebih besar dari kuota, Anda dapat mengatur kebijakan Anda ke dalam beberapa pernyataan dan mengelompokkan pernyataan ke dalam 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 IAM JSON](#)

AWS kebijakan terkelola dengan peringatan umum ini

[AWS kebijakan terkelola](#) memungkinkan Anda memulai AWS dengan menetapkan izin berdasarkan kasus AWS penggunaan umum.

Kebijakan AWS terkelola 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

Dalam AWS Management Console, 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 atau, 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

Kebijakan berbasis sumber daya adalah dokumen kebijakan JSON yang Anda lampirkan ke sumber daya, seperti bucket Amazon S3. Kebijakan ini memberikan izin pokok yang ditentukan untuk melakukan tindakan spesifik pada sumber daya tersebut dan menentukan dalam kondisi apa hal ini berlaku. Ukuran kebijakan berbasis sumber daya tidak dapat melebihi kuota yang ditetapkan untuk sumber daya tersebut. Kuota, juga disebut sebagai batas dalam 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 Anda ke dalam beberapa pernyataan dan mengelompokkan pernyataan ke dalam beberapa kebijakan.

Istilah terkait

- [Kebijakan berbasis sumber daya](#)
- [Kebijakan bucket Amazon S3](#)
- [Beberapa pernyataan dan beberapa kebijakan](#)

- [Ikhtisar kebijakan JSON](#)
- [Tata bahasa kebijakan IAM JSON](#)

Peringatan Umum - Ketidakcocokan Jenis

Dalam AWS Management Console, 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 atau, 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 IAM JSON: Operator kondisi](#)

Peringatan Umum - Ketik ketidakcocokan Boolean

Dalam AWS Management Console, 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 atau, 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 IAM JSON: Operator kondisi](#)

Peringatan Umum - Ketik tanggal ketidakcocokan

Dalam AWS Management Console, 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 atau, 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 ISO 8601 lainnya.

Istilah terkait

- [Operator kondisi tanggal](#)
- [Elemen kebijakan IAM JSON: Operator kondisi](#)

Peringatan Umum - Ketik nomor ketidakcocokan

Dalam AWS Management Console, 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 atau, 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 IAM JSON: Operator kondisi](#)

Peringatan Umum - Ketik string ketidakcocokan

Dalam AWS Management Console, 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 atau, 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 IAM JSON: Operator kondisi](#)

Peringatan Umum - Direkomendasikan repo dan cabang github khusus

Dalam AWS Management Console, 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.
```

Dalam panggilan terprogram ke AWS API AWS CLI atau, 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."
```

Menyelesaikan peringatan umum

Jika Anda menggunakan GitHub sebagai IDP OIDC, praktik terbaik adalah membatasi entitas yang dapat mengambil peran yang terkait dengan IDP IAM. Saat menyertakan Condition pernyataan dalam kebijakan kepercayaan peran, Anda dapat membatasi peran tersebut ke GitHub organisasi, repositori, atau cabang tertentu. Anda dapat menggunakan tombol kondisi `token.actions.githubusercontent.com:sub` untuk membatasi akses. Kami menyarankan Anda membatasi kondisi untuk satu set repositori atau cabang tertentu. Jika Anda menggunakan wildcard (*) di `token.actions.githubusercontent.com:sub`, maka GitHub Tindakan dari organisasi atau repositori di luar kendali Anda dapat mengambil peran yang terkait dengan GitHub IDP IAM di akun Anda. AWS

Istilah terkait

- [Mengkonfigurasi peran untuk penyedia identitas GitHub OIDC](#)

Peringatan Umum - Ukuran kebijakan melebihi kuota kebijakan kepercayaan peran

Dalam AWS Management Console, 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 atau, 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 putih, melebihi maksimum karakter. Kami menyarankan Anda untuk meminta peningkatan kuota kebijakan kepercayaan peran menggunakan Service Quotas dan. AWS Support Center Console

Istilah terkait

- [IAM dan AWS STS kuota, persyaratan nama, dan batas karakter](#)

Peringatan Keamanan — Izinkan dengan NotPrincipal

Dalam AWS Management Console, 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 atau, 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 elemen Principal. 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

Dalam AWS Management Console, 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 atau, 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 multivaluasi. Operator ForAllValues set menguji apakah nilai setiap anggota dari set permintaan adalah subset dari set 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 multivalued. Misalnya, Amazon SES mendukung kunci dengan nilai tunggal (String) dan tipe data ArrayOfString multivalued.

- [Kunci konteks multivaluasi](#)

Peringatan Keamanan - Lulus peran dengan NotResource

Dalam AWS Management Console, 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 atau, 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 mengonfigurasi banyak AWS layanan, Anda harus meneruskan peran IAM ke layanan. Untuk mengizinkan ini, Anda harus memberikan `iam:PassRole` izin kepada identitas (pengguna, grup pengguna, atau peran). Menggunakan `iam:PassRole` kebijakan dengan `NotResource` elemen dapat memungkinkan prinsipal Anda mengakses lebih banyak layanan atau fitur daripada yang Anda inginkan. AWS merekomendasikan agar Anda menentukan ARN yang diizinkan dalam `Resource` elemen sebagai gantinya. Selain itu, Anda dapat mengurangi izin ke satu layanan dengan menggunakan tombol `iam:PassedToService` kondisi.

- [Melewati peran ke layanan](#)
- [saya: PassedToService](#)
- [Elemen kebijakan IAM JSON: NotResource](#)
- [Elemen kebijakan IAM JSON: Sumber daya](#)

Peringatan Keamanan - Lulus peran dengan bintang beraksi dan NotResource

Dalam AWS Management Console, 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 atau, 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 mengonfigurasi banyak AWS layanan, Anda harus meneruskan peran IAM ke layanan. Untuk mengizinkan ini, Anda harus memberikan `iam:PassRole` izin kepada identitas (pengguna, grup pengguna, atau peran). Kebijakan dengan wildcard (*) dalam Action dan yang menyertakan `NotResource` elemen dapat memungkinkan prinsipal Anda mengakses lebih banyak layanan atau fitur daripada yang Anda inginkan. AWS merekomendasikan agar Anda menentukan ARN yang diizinkan dalam Resource elemen sebagai gantinya. Selain itu, Anda dapat mengurangi izin ke satu layanan dengan menggunakan tombol `iam:PassedToService` kondisi.

- [Melewati peran ke layanan](#)
- [saya: PassedToService](#)
- [Elemen kebijakan IAM JSON: NotResource](#)
- [Elemen kebijakan IAM JSON: Sumber daya](#)

Peringatan Keamanan - Lulus peran dengan NotAction dan NotResource

Dalam AWS Management Console, 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 atau, 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 mengonfigurasi banyak AWS layanan, Anda harus meneruskan peran IAM ke layanan. Untuk mengizinkan ini, Anda harus memberikan `iam:PassRole` izin kepada identitas (pengguna, grup pengguna, atau peran). Menggunakan `NotAction` elemen dan daftar beberapa sumber daya dalam `NotResource` elemen dapat memungkinkan prinsipal Anda untuk mengakses lebih banyak layanan atau fitur daripada yang Anda inginkan. AWS merekomendasikan agar Anda menentukan ARN yang diizinkan dalam `Resource` elemen sebagai gantinya. Selain itu, Anda dapat mengurangi izin ke satu layanan dengan menggunakan tombol `iam:PassedToService` kondisi.

- [Melewati peran ke layanan](#)
- [saya: PassedToService](#)
- [Elemen kebijakan IAM JSON: NotAction](#)
- [Elemen kebijakan IAM JSON: Tindakan](#)
- [Elemen kebijakan IAM JSON: NotResource](#)
- [Elemen kebijakan IAM JSON: Sumber daya](#)

Peringatan Keamanan - Lulus peran dengan bintang di sumber daya

Dalam AWS Management Console, 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 atau, 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 mengonfigurasi banyak AWS layanan, Anda harus meneruskan peran IAM ke layanan. Untuk mengizinkan ini, Anda harus memberikan `iam:PassRole` izin kepada identitas (pengguna, grup pengguna, atau peran). Kebijakan yang memungkinkan `iam:PassRole` dan yang menyertakan

wildcard (*) dalam Resource elemen dapat memungkinkan prinsipal Anda mengakses lebih banyak layanan atau fitur daripada yang Anda inginkan. AWS merekomendasikan agar Anda menentukan ARN yang diizinkan dalam Resource elemen sebagai gantinya. Selain itu, Anda dapat mengurangi izin ke satu layanan dengan menggunakan tombol `iam:PassedToService` kondisi.

Beberapa AWS layanan menyertakan namespace layanan mereka atas nama peran mereka. Pemeriksaan kebijakan ini mempertimbangkan konvensi ini saat menganalisis kebijakan untuk menghasilkan temuan. Misalnya, ARN sumber daya berikut mungkin tidak menghasilkan temuan:

```
arn:aws:iam::*:role/Service*
```

- [Melewati peran ke layanan](#)
- [saya: PassedToService](#)
- [Elemen kebijakan IAM JSON: Sumber daya](#)

AWS kebijakan terkelola dengan peringatan keamanan ini

[AWS kebijakan terkelola](#) memungkinkan Anda memulai AWS dengan menetapkan izin berdasarkan kasus AWS penggunaan umum.

Salah satu kasus penggunaan tersebut adalah untuk administrator dalam akun Anda. Kebijakan AWS terkelola berikut menyediakan akses administrator dan memberikan izin untuk meneruskan peran IAM apa pun ke layanan apa pun. AWS merekomendasikan agar Anda melampirkan kebijakan AWS terkelola berikut hanya ke identitas IAM yang Anda anggap administrator.

- [AdministratorAccess-Memperkuat](#)

Kebijakan AWS terkelola berikut mencakup izin `iam:PassRole` dengan wildcard (*) di sumber daya dan berada di jalur [penghentian](#). Untuk setiap kebijakan ini, kami memperbarui panduan izin, seperti merekomendasikan kebijakan AWS terkelola baru yang mendukung kasus penggunaan. Untuk melihat alternatif kebijakan ini, lihat panduan untuk [setiap layanan](#).

- `AWSElasticBeanstalkFullAccess`
- `AWSElasticBeanstalkService`
- `AWSLambdaFullAccess`
- `AWSLambdaReadOnlyAccess`

- `AWSOpsWorksFullAccess`
- `AWSOpsWorksRole`
- `AWSDataPipelineRole`
- `AmazonDynamoDB FullAccesswithDataPipeline`
- `AmazonElasticMapReduceFullAccess`
- `AmazonDynamoDB FullAccesswithDataPipeline`
- `AmazonEC2 ContainerServiceFullAccess`

Kebijakan AWS terkelola 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

Dalam AWS Management Console, temukan 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 atau, temukan 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 mengonfigurasi banyak AWS layanan, Anda harus meneruskan peran IAM ke layanan. Untuk mengizinkan ini, Anda harus memberikan `iam:PassRole` izin kepada identitas (pengguna, grup pengguna, atau peran). Kebijakan dengan wildcard (*) di Resource elemen Action dan

dapat memungkinkan prinsipal Anda mengakses lebih banyak layanan atau fitur daripada yang Anda inginkan. AWS merekomendasikan agar Anda menentukan ARN yang diizinkan dalam Resource elemen sebagai gantinya. Selain itu, Anda dapat mengurangi izin ke satu layanan dengan menggunakan tombol `iam:PassedToService` kondisi.

- [Melewati peran ke layanan](#)
- [saya: PassedToService](#)
- [Elemen kebijakan IAM JSON: Tindakan](#)
- [Elemen kebijakan IAM JSON: Sumber daya](#)

AWS kebijakan terkelola dengan peringatan keamanan ini

[AWS kebijakan terkelola](#) memungkinkan Anda memulai AWS dengan menetapkan izin berdasarkan kasus AWS penggunaan umum.

Beberapa kasus penggunaan tersebut adalah untuk administrator dalam akun Anda. Kebijakan AWS terkelola berikut menyediakan akses administrator dan memberikan izin untuk meneruskan peran IAM apa pun ke layanan apa pun AWS . AWS merekomendasikan agar Anda melampirkan kebijakan AWS terkelola berikut hanya pada identitas IAM yang Anda anggap administrator.

- [AdministratorAccess](#)
- [IAM FullAccess](#)

Peringatan Keamanan - Lulus peran dengan bintang di sumber daya dan NotAction

Dalam AWS Management Console, 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 atau, 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 mengonfigurasi banyak AWS layanan, Anda harus meneruskan peran IAM ke layanan. Untuk mengizinkan ini, Anda harus memberikan iam:PassRole izin kepada identitas (pengguna, grup pengguna, atau peran). Menggunakan NotAction elemen dalam kebijakan dengan wildcard (*) dalam Resource elemen dapat memungkinkan prinsipal Anda mengakses lebih banyak layanan atau fitur daripada yang Anda inginkan. AWS merekomendasikan agar Anda menentukan ARN yang diizinkan dalam Resource elemen sebagai gantinya. Selain itu, Anda dapat mengurangi izin ke satu layanan dengan menggunakan tombol iam:PassedToService kondisi.

- [Melewati peran ke layanan](#)
- [saya: PassedToService](#)
- [Elemen kebijakan IAM JSON: NotAction](#)
- [Elemen kebijakan IAM JSON: Tindakan](#)
- [Elemen kebijakan IAM JSON: Sumber daya](#)

Peringatan Keamanan - Kunci kondisi berpasangan tidak ada

Dalam AWS Management Console, 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 atau, 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 tombol kondisi lebih aman saat dipasangkan dengan kunci kondisi terkait lainnya. AWS merekomendasikan agar Anda menyertakan kunci kondisi terkait di blok kondisi yang sama dengan kunci kondisi yang ada. Hal ini membuat izin yang diberikan melalui kebijakan lebih aman.

Misalnya, Anda dapat menggunakan tombol `aws:VpcSourceIp` kondisi untuk membandingkan alamat IP dari mana permintaan dibuat dengan alamat IP yang Anda tentukan dalam kebijakan. AWS merekomendasikan 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

Dalam AWS Management Console, 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 atau, 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 diabaikan untuk layanan tersebut. AWS merekomendasikan 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 tidak termasuk 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.

Saat layanan mendukung kunci `aws:ResourceTag` kondisi, Anda dapat menggunakan tag untuk mengontrol akses ke sumber daya layanan tersebut. Ini dikenal sebagai [Attribute-based Access Control \(ABAC\)](#). Layanan yang tidak mendukung kunci ini mengharuskan Anda untuk 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 `aws:ResourceTag` subset 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 bahwa 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 awalan. "Confidential" Kemudian sertakan pernyataan terpisah yang mencegah penghapusan sumber daya dengan konvensi penamaan itu.

```
{
  "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 operator kondisi sebagai solusi untuk temuan ini. Ini berarti “Tolak tindakan jika kunci ada dalam konteks permintaan dan nilainya cocok. Kalau tidak, tolak tindakannya.” Pada contoh sebelumnya, termasuk `lambda:DeleteFunction` tindakan dalam `DenyDeleteSupported` pernyataan dengan `StringEqualsIfExists` operator selalu menyangkal tindakan. Untuk tindakan itu, kuncinya 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 dengan RBAC](#)
- [Elemen kebijakan IAM JSON: Operator kondisi](#)
- [Elemen kondisi](#)
- [Ikhtisar kebijakan JSON](#)

Peringatan Keamanan - Tolak NotAction dengan kunci kondisi tag yang tidak didukung untuk layanan

Dalam AWS Management Console, 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 atau, 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 ini 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 merekomendasikan 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](#).

Saat layanan mendukung kunci `aws:ResourceTag` kondisi, Anda dapat menggunakan tag untuk mengontrol akses ke sumber daya layanan tersebut. Ini dikenal sebagai [Attribute-based Access Control \(ABAC\)](#). Layanan yang tidak mendukung kunci ini mengharuskan Anda untuk mengontrol akses ke sumber daya menggunakan [kontrol akses berbasis sumber daya \(RBAC\)](#).

Istilah terkait

- [Kunci kondisi global](#)

- [Membandingkan ABAC dengan RBAC](#)
- [Elemen kebijakan IAM JSON: Operator kondisi](#)
- [Elemen kondisi](#)
- [Ikhtisar kebijakan JSON](#)

Peringatan Keamanan — Batasi akses ke prinsipal layanan

Dalam AWS Management Console, 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 atau, 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 pengenal untuk layanan. Saat memberikan akses ke kepala layanan untuk bertindak atas nama Anda, batasi akses. Anda dapat mencegah kebijakan yang terlalu permisif dengan menggunakan `aws:SourceArn`, `aws:SourceAccount`, `aws:SourceOrgID`, atau kunci `aws:SourceOrgPaths` kondisi untuk membatasi akses ke sumber tertentu, seperti ARN sumber daya tertentu, ID organisasi Akun AWS, atau jalur organisasi. Membatasi akses membantu Anda mencegah masalah keamanan yang disebut masalah wakil yang bingung.

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 tidak ada untuk kepala sekolah oidc

Dalam AWS Management Console, 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 atau, 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 prinsipal Open ID Connect tanpa kondisi bisa terlalu permisif. Tambahkan kunci kondisi dengan awalan yang cocok dengan prinsip 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 tidak ada

Dalam AWS Management Console, 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 atau, 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. Saat menyertakan Condition pernyataan dalam kebijakan kepercayaan peran, Anda dapat membatasi peran tersebut ke GitHub organisasi, repositori, atau cabang tertentu. Anda dapat menggunakan tombol kondisi `token.actions.githubusercontent.com:sub` untuk membatasi akses. Kami menyarankan Anda membatasi kondisi untuk satu set repositori atau cabang tertentu. Jika Anda tidak menyertakan kondisi ini, maka GitHub Tindakan 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](#)

Saran - Tindakan array kosong

Dalam AWS Management Console, 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 atau, 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 salah satu Action atau NotAction elemen yang mencakup serangkaian tindakan. Ketika elemen kosong, pernyataan kebijakan tidak memberikan izin. Tentukan tindakan dalam Action elemen.

- [Elemen kebijakan IAM JSON: Tindakan](#)

Saran - Kondisi array kosong

Dalam AWS Management Console, 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 atau, 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 kembali true dan pernyataan kebijakan tidak memberikan izin. Tentukan nilai kondisi.

- [Elemen kebijakan IAM JSON: Kondisi](#)

Saran - Kondisi array kosong ForAllValues

Dalam AWS Management Console, 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 atau, 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 dari set permintaan adalah subset dari set kunci kondisi.

Bila 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 multivaluasi](#)
- [Operator kondisi nol](#)
- [Elemen kebijakan IAM JSON: Kondisi](#)

Saran - Kondisi array kosong `ForAnyValue`

Dalam AWS Management Console, 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 atau, 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 kumpulan nilai permintaan cocok dengan setidaknya satu anggota dari kumpulan nilai kunci kondisi.

Bila Anda menggunakan `ForAnyValues` dengan kunci kondisi kosong, kondisi tidak pernah cocok. Ini berarti bahwa pernyataan tersebut tidak berpengaruh pada kebijakan tersebut. AWS merekomendasikan agar Anda menulis ulang kondisinya.

- [Kunci konteks multivaluasi](#)
- [Elemen kebijakan IAM JSON: Kondisi](#)

Saran - Kondisi array kosong IfExists

Dalam AWS Management Console, 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 atau, 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

...IfExistsSufiks 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.

Bila 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 IAM JSON: Kondisi](#)

Saran - Prinsipal array kosong

Dalam AWS Management Console, 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 atau, 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. Kebijakan berbasis sumber daya adalah kebijakan yang diterapkan langsung ke sumber daya.

Bila Anda menyediakan array kosong dalam `Principal` elemen pernyataan, pernyataan tersebut tidak berpengaruh pada kebijakan. AWS merekomendasikan agar Anda menentukan prinsip yang harus memiliki akses ke sumber daya.

- [Elemen kebijakan IAM JSON: Principal](#)
- [Elemen kebijakan IAM JSON: NotPrincipal](#)

Saran - Sumber daya array kosong

Dalam AWS Management Console, 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 atau, 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 menyediakan array kosong dalam elemen sumber daya pernyataan, pernyataan tersebut tidak berpengaruh pada kebijakan. AWS merekomendasikan agar Anda menentukan Nama Sumber Daya Amazon (ARN) untuk sumber daya.

- [Elemen kebijakan IAM JSON: Sumber daya](#)
- [Elemen kebijakan IAM JSON: NotResource](#)

Saran - Kondisi objek kosong

Dalam AWS Management Console, 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 atau, 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.

Ketika Anda memberikan objek kosong dalam elemen kondisi pernyataan, pernyataan tersebut tidak berpengaruh pada kebijakan. Hapus elemen opsional atau tentukan kondisi.

- [Elemen kebijakan IAM JSON: Kondisi](#)

Saran - Prinsip objek kosong

Dalam AWS Management Console, 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 atau, 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. Kebijakan berbasis sumber daya adalah kebijakan yang diterapkan langsung ke sumber daya.

Ketika Anda memberikan objek kosong dalam `Principal` elemen pernyataan, pernyataan tersebut tidak berpengaruh pada kebijakan. AWS merekomendasikan agar Anda menentukan prinsip yang harus memiliki akses ke sumber daya.

- [Elemen kebijakan IAM JSON: Principal](#)
- [Elemen kebijakan IAM JSON: NotPrincipal](#)

Saran - Nilai Sid kosong

Dalam AWS Management Console, 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 atau, 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 pengenal 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 IAM JSON: Sid](#)

Saran - Tingkatkan rentang IP

Dalam AWS Management Console, 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 atau, 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. Ketika Anda memasukkan bit bukan nol setelah bit bertopeng, mereka tidak dipertimbangkan untuk kondisi tersebut. AWS merekomendasikan agar Anda menggunakan alamat baru yang disertakan dalam pesan.

- [Operator kondisi alamat IP](#)
- [Elemen kebijakan IAM JSON: Kondisi](#)

Saran - Null dengan kualifikasi

Dalam AWS Management Console, 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 atau, 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 satu kunci kondisi, Anda harus menggunakan `ForAllValues` atau `ForAnyValue` mengatur operator.

Bila Anda menggunakan operator `Null` kondisi dengan `ForAllValues`, pernyataan selalu kembali `true`. Bila 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 multivaluasi](#)
- [Operator kondisi nol](#)
- [Elemen kondisi](#)

Saran - Subset alamat IP pribadi

Dalam AWS Management Console, 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 atau, 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 IAM JSON: Kondisi](#)

Saran - Subset pribadi NotIpAddress

Dalam AWS Management Console, 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 atau, 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` dengan `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 IAM JSON: Kondisi](#)

Saran — Tindakan berlebihan

Dalam AWS Management Console, 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 atau, 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

Saat Anda menggunakan wildcard (*) dalam Action elemen, Anda dapat menyertakan izin yang 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 termasuk `iam:GetCredentialReport` tindakan dua kali.

```
"Action": [  
  "iam:Get*",  
  "iam:List*",  
  "iam:GetCredentialReport"  
],
```

Dalam contoh ini, izin didefinisikan untuk setiap tindakan IAM yang dimulai dengan `Get` atau `List`. Ketika IAM menambahkan operasi `get` atau `list` tambahan, kebijakan ini akan mengizinkannya. Anda mungkin ingin mengizinkan semua tindakan hanya-baca ini. `iam:GetCredentialReport` Tindakan ini sudah termasuk sebagai bagian dari `iam:Get*`. Untuk menghapus izin duplikat, Anda dapat menghapus `iam:GetCredentialReport`.

Anda menerima temuan untuk pemeriksaan kebijakan ini ketika 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 salah satu `iam:Get*` atau `iam:*CredentialReport` akan mengubah izin kebijakan.

- [Elemen kebijakan IAM JSON: Tindakan](#)

AWS kebijakan terkelola dengan saran ini

[AWS kebijakan terkelola](#) 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 sarankan agar Anda menghapus tindakan berlebihan dari kebijakan Anda.

Saran — Nilai kondisi redundan num

Dalam AWS Management Console, 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 atau, 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 usia tumpang tindih 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 nilai redundan. 2700

- [Operator kondisi numerik](#)
- [Elemen kebijakan IAM JSON: Kondisi](#)

Saran — Sumber daya redundan

Dalam AWS Management Console, 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 atau, 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 Anda 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 `jane-admin` redundan dan `jane-s3only` ARN tanpa mengubah izin yang dihasilkan. Hal ini membuat kebijakan menjadi dinamis. Ini akan menentukan izin untuk setiap peran masa depan yang dimulai dengan `jane`. Jika tujuan kebijakan ini adalah untuk mengizinkan akses ke sejumlah peran statis, hapus ARN terakhir dan daftarkan hanya ARN yang harus ditentukan.

- [Elemen kebijakan IAM JSON: Sumber daya](#)

AWS kebijakan terkelola dengan saran ini

[AWS kebijakan terkelola](#) 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, AWS sarankan Anda menghapus sumber daya yang berlebihan dari kebijakan Anda.

Saran — Pernyataan berlebihan

Dalam AWS Management Console, 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 atau, 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 duplikat. Ini mungkin menghasilkan lebih banyak izin daripada yang dimaksudkan.

- [Elemen kebijakan IAM JSON: Pernyataan](#)

Saran — Wildcard dalam nama layanan

Dalam AWS Management Console, 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 atau, 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

Ketika Anda memasukkan nama AWS layanan dalam kebijakan, AWS merekomendasikan agar Anda tidak menyertakan wildcard (*,?). Ini mungkin menambahkan izin untuk layanan future yang tidak Anda inginkan. Misalnya, ada lebih dari selusin AWS layanan dengan kata `*code*` dalam nama mereka.

```
"Resource": "arn:aws:*code*::111122223333:*"
```

- [Elemen kebijakan IAM JSON: Sumber daya](#)

Saran - Izinkan dengan kunci kondisi tag yang tidak didukung untuk layanan

Dalam AWS Management Console, 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 atau, 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 dalam `Condition` elemen kebijakan dengan `"Effect": "Allow"` tidak memengaruhi izin yang diberikan oleh kebijakan, karena kondisi diabaikan untuk tindakan layanan tersebut. AWS merekomendasikan 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 tidak termasuk 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.

Saat layanan mendukung kunci `aws:ResourceTag` kondisi, Anda dapat menggunakan tag untuk mengontrol akses ke sumber daya layanan tersebut. Ini dikenal sebagai [Attribute-based Access Control \(ABAC\)](#). Layanan yang tidak mendukung kunci ini mengharuskan Anda untuk 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 `aws:ResourceTag` subset 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 mengizinkan anggota tim untuk melihat detail sumber daya tertentu yang ditandai dengan pasangan nilai kunci. `team=BumbleBee` Juga asumsikan bahwa 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 kunci `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 itu.

```
{
  "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, termasuk `lambda:GetFunction` tindakan dalam `AllowViewSupported` pernyataan dengan `StringNotEquals` operator selalu memungkinkan tindakan, terlepas dari apakah sumber daya ditandai.

Jangan gunakan... [IfExists](#) versi operator kondisi sebagai solusi untuk temuan ini. Ini berarti "Izinkan tindakan jika kunci ada dalam konteks permintaan dan nilainya cocok. Jika tidak, izinkan aksinya." Dalam contoh sebelumnya, termasuk `lambda:GetFunction` tindakan dalam `AllowViewSupported` pernyataan dengan `StringEqualsIfExists` operator selalu memungkinkan tindakan. Untuk tindakan itu, kuncinya tidak ada dalam konteks, dan setiap upaya untuk melihat jenis sumber daya diizinkan, terlepas dari apakah sumber daya diberi tag.

Istilah terkait

- [Kunci kondisi global](#)
- [Elemen kebijakan IAM JSON: Operator kondisi](#)
- [Elemen kondisi](#)

- [Ikhtisar kebijakan JSON](#)

Saran - Izinkan NotAction dengan kunci kondisi tag yang tidak didukung untuk layanan

Dalam AWS Management Console, 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 atau, 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 dalam Condition elemen kebijakan dengan elemen NotAction dan "Effect": "Allow" tidak memengaruhi izin yang diberikan oleh kebijakan. Kondisi ini 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 diperbolehkan. AWS merekomendasikan 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](#).

Saat layanan mendukung kunci `aws:ResourceTag` kondisi, Anda dapat menggunakan tag untuk mengontrol akses ke sumber daya layanan tersebut. Ini dikenal sebagai [Attribute-based Access Control \(ABAC\)](#). Layanan yang tidak mendukung kunci ini mengharuskan Anda untuk mengontrol akses ke sumber daya menggunakan [kontrol akses berbasis sumber daya \(RBAC\)](#).

Istilah terkait

- [Kunci kondisi global](#)
- [Membandingkan ABAC dengan RBAC](#)
- [Elemen kebijakan IAM JSON: Operator kondisi](#)
- [Elemen kondisi](#)
- [Ikhtisar kebijakan JSON](#)

Saran - Kunci kondisi yang direkomendasikan untuk prinsipal layanan

Dalam AWS Management Console, 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 atau, 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 pengenal untuk layanan. Anda harus menggunakan `aws:SourceArn`, `aws:SourceAccount`, `aws:SourceOrgID`, atau kunci `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 bingung.

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

Dalam AWS Management Console, 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 atau, 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 jalur akses Amazon S3.

Anda harus menggunakan kunci kondisi dalam kebijakan berbasis identitas untuk mengontrol akses ke sumber daya.

Istilah terkait

- [Kebijakan berbasis identitas dan kebijakan berbasis sumber daya](#)

Saran — Prinsip redundan dalam kebijakan kepercayaan peran

Dalam AWS Management Console, 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 atau, 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 prinsipal peran yang diasumsikan dan peran induknya dalam `Principal` elemen kebijakan, kebijakan tersebut 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"
  ]
}
```

Kami merekomendasikan untuk menghapus prinsipal peran yang diasumsikan.

Istilah terkait

- [Kepala sesi peran](#)

Saran — Konfirmasikan jenis klaim audiens

Dalam AWS Management Console, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Confirm audience claim type: The 'aud' (audience) claim key identifies the recipients that the JSON web token is intended for. Audience claims can be multivalued or single-valued. If the claim is multivalued, use a ForAllValues or ForAnyValue qualifier. If the claim is single-valued, do not use a qualifier.
```

Dalam panggilan terprogram ke AWS API AWS CLI atau, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The 'aud' (audience) claim key identifies the recipients that the JSON web token is intended for. Audience claims can be multivalued or single-valued. If the claim is multivalued, use a ForAllValues or ForAnyValue qualifier. If the claim is single-valued, do not use a qualifier."
```

Menyelesaikan saran

Kunci klaim aud (audiens) adalah pengenalan unik untuk aplikasi Anda yang diberikan kepada Anda saat Anda mendaftarkan aplikasi dengan iDP dan mengidentifikasi penerima yang dimaksudkan untuk token web JSON. Klaim audiens dapat bernilai multivaluasi atau bernilai tunggal. Jika klaim multivaluasi, gunakan operator set `ForAllValues` atau `ForAnyValue` kondisi. Jika klaim bernilai tunggal, jangan gunakan operator set kondisi.

Istilah terkait

- [Membuat peran untuk identitas web atau OpenID Connect Federation \(konsol\)](#)
- [Kunci konteks multivaluasi](#)
- [Kunci kondisi bernilai tunggal vs. multivaluasi](#)

Pemeriksaan kebijakan khusus IAM Access Analyzer

Anda dapat memvalidasi kebijakan Anda terhadap standar keamanan yang ditentukan menggunakan pemeriksaan kebijakan AWS Identity and Access Management Access Analyzer khusus. Ada dua jenis pemeriksaan kebijakan khusus yang dapat Anda jalankan:

- **Memeriksa kebijakan referensi:** Saat mengedit kebijakan, Anda dapat memeriksa apakah kebijakan yang diperbarui memberikan akses baru dibandingkan dengan kebijakan referensi, seperti versi kebijakan yang 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.
- **Periksa daftar tindakan IAM:** Anda dapat memeriksa untuk memastikan bahwa tindakan IAM tertentu tidak diizinkan oleh kebijakan Anda. Anda dapat menjalankan pemeriksaan ini saat membuat atau mengedit kebijakan menggunakan AWS CLI atau API.

Biaya terkait dengan setiap pemeriksaan kebijakan khusus. Untuk detail selengkapnya tentang harga, lihat harga [IAM Access Analyzer](#).

Cara kerja pemeriksaan kebijakan khusus

Anda dapat menjalankan pemeriksaan kebijakan khusus pada identitas dan kebijakan berbasis sumber daya. Pemeriksaan kebijakan khusus tidak bergantung pada teknik pencocokan pola atau memeriksa log akses untuk menentukan apakah akses baru atau yang ditentukan diizinkan oleh kebijakan. Mirip dengan temuan akses eksternal, pemeriksaan kebijakan khusus dibangun

di [Zelkova](#). Zelkova menerjemahkan kebijakan IAM ke dalam pernyataan logis yang setara, dan menjalankan serangkaian pemecah logis tujuan umum dan khusus (teori modulo kepuasan) terhadap masalah. Untuk memeriksa akses baru atau yang ditentukan, IAM Access Analyzer menerapkan Zelkova berulang kali ke kebijakan. Pertanyaan menjadi semakin spesifik untuk mengkarakterisasi kelas-kelas perilaku yang diizinkan oleh kebijakan berdasarkan konten kebijakan. Untuk informasi lebih lanjut tentang teori modulo kepuasan, lihat Teori Modulo [Kepuasan](#).

Dalam kasus yang jarang terjadi, IAM Access Analyzer tidak dapat sepenuhnya menentukan apakah pernyataan kebijakan memberikan akses baru atau yang ditentukan. Dalam kasus tersebut, kesalahan di sisi mendeklarasikan positif palsu dengan gagal 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 untuk kebijakan referensi dan mempelajari cara menyiapkan dan menjalankan pemeriksaan kebijakan kustom untuk akses baru dalam kebijakan kustom [IAM Access Analyzer memeriksa repositori sampel](#) pada GitHub

Sebelum menggunakan contoh-contoh ini

Sebelum Anda menggunakan contoh kebijakan referensi ini, lakukan hal berikut:

- Tinjau dan sesuaikan kebijakan referensi dengan cermat untuk kebutuhan unik Anda.
- Uji kebijakan referensi di lingkungan Anda secara menyeluruh dengan Layanan AWS yang Anda gunakan.

Kebijakan referensi menunjukkan implementasi dan penggunaan pemeriksaan kebijakan khusus. 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 lingkungan dalam analisisnya. Analisis mereka hanya mempertimbangkan informasi yang terkandung dalam kebijakan input. Misalnya, pemeriksaan kebijakan khusus 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

Jika pemeriksaan kebijakan kustom gagal, respons dari IAM Access Analyzer menyertakan [ID pernyataan \(Sid\)](#) pernyataan kebijakan yang menyebabkan pemeriksaan gagal. Meskipun ID pernyataan adalah elemen kebijakan opsional, sebaiknya Anda menambahkan ID pernyataan untuk setiap pernyataan kebijakan. Pemeriksaan kebijakan kustom juga menampilkan indeks pernyataan untuk membantu mengidentifikasi alasan kegagalan pemeriksaan. Indeks pernyataan mengikuti penomoran berbasis nol, di mana pernyataan pertama direferensikan sebagai 0. Ketika ada beberapa pernyataan yang menyebabkan cek gagal, cek hanya mengembalikan satu ID pernyataan pada satu waktu. Kami menyarankan Anda memperbaiki pernyataan yang disorot dalam alasan dan menjalankan kembali cek sampai lulus.

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 AWS Management Console 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 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](#).

Pembuatan kebijakan IAM Access Analyzer

Sebagai administrator atau pengembang, Anda dapat memberikan izin kepada entitas IAM (pengguna atau peran) di luar 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 suatu entitas. IAM Access Analyzer meninjau AWS CloudTrail log Anda dan menghasilkan templat kebijakan yang berisi izin yang digunakan entitas dalam rentang tanggal yang ditentukan. 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](#)
- [Menghasilkan kebijakan berdasarkan CloudTrail aktivitas \(AWS CLI\)](#)
- [Menghasilkan 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). Ini 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 jangka waktu hingga 90 hari untuk IAM Access Analyzer untuk menganalisis peristiwa historis AWS CloudTrail Anda. Anda harus menentukan peran layanan yang sudah ada atau membuat peran layanan baru. Peran layanan memberikan IAM Access Analyzer akses ke CloudTrail jejak Anda dan informasi 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 [IAM Access Analyzer](#).
- Buat kebijakan — IAM Access Analyzer menghasilkan kebijakan berdasarkan aktivitas akses di acara Anda CloudTrail .
- 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

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, lihat [Layanan 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 AWS Management Console, 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.

- Aktifkan CloudTrail jejak — Anda harus mengaktifkan CloudTrail jejak untuk akun Anda untuk membuat kebijakan berdasarkan aktivitas akses. Saat Anda membuat CloudTrail jejak, CloudTrail kirimkan 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.

- `PassRoleiam:PassRole`Tindakan ini tidak dilacak oleh CloudTrail dan tidak termasuk dalam kebijakan yang dihasilkan.
- Mengurangi waktu pembuatan kebijakan — Untuk menghasilkan kebijakan lebih cepat, kurangi rentang tanggal yang Anda tentukan selama penyiapan pembuatan kebijakan.
- Gunakan CloudTrail untuk audit - Jangan gunakan pembuatan kebijakan untuk tujuan audit; gunakan CloudTrail sebagai gantinya. Untuk informasi selengkapnya tentang penggunaan CloudTrail, lihat [Mencatat panggilan IAM dan AWS STS API dengan AWS CloudTrail](#).
- Tindakan yang ditolak — Pembuatan kebijakan meninjau semua CloudTrail peristiwa, termasuk tindakan yang ditolak.
- Satu konsol IAM kebijakan — Anda dapat memiliki satu kebijakan yang dihasilkan pada satu waktu di konsol IAM.
- Konsol IAM ketersediaan kebijakan yang dihasilkan — 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 IAM Access Analyzer, lihat kuota [IAM](#) Access Analyzer.
- 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 mengenakan 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 AWS Control Tower pembuatan kebijakan.

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.

Penyiapan pertama kali

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 memberikan IAM Access Analyzer akses ke 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 layanan. AWS](#)

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 membuat kebijakan. Saat Anda menggunakan konsol IAM untuk membuat peran layanan sebagai bagian dari proses pembuatan kebijakan, kami membuat kebijakan ini untuk Anda.

```
{
  "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:::DOC-EXAMPLE-BUCKET",
        "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
      ]
    }
  ]
}
```

```
    ]
  }
]
}
```

Contoh kebijakan berikut menunjukkan kebijakan kepercayaan peran dengan izin yang memungkinkan IAM Access Analyzer untuk mengambil peran.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "access-analyzer.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Penggunaan selanjutnya

Untuk membuat kebijakan di AWS Management Console, pengguna IAM harus memiliki kebijakan izin yang memungkinkan mereka meneruskan peran layanan yang digunakan untuk pembuatan kebijakan ke IAM Access Analyzer. `iam:PassRole` biasanya disertai dengan `iam:GetRole` agar pengguna bisa mendapatkan detail 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 layanan AWS](#)

```
{
  "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 IAM Access Analyzer berikut untuk membuat kebijakan di AWS API, AWS Management Console, 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

Ketika Anda menggunakan AWS Management Console untuk membuat kebijakan, Anda harus memiliki `cloudtrail:ListTrails` izin untuk membuat daftar 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 AWS Management Console.

Menghasilkan kebijakan untuk peran IAM

1. Masuk ke AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Pada panel navigasi di sebelah kiri, pilih Peran.

Note

Langkah-langkah untuk menghasilkan 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 Buat kebijakan berdasarkan CloudTrail peristiwa, pilih Buat kebijakan.
5. Pada halaman Buat kebijakan, tentukan periode waktu yang Anda inginkan IAM Access Analyzer untuk 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 ada yang sesuai atau buat peran baru jika peran yang sesuai tidak ada. Peran tersebut memberikan izin IAM Access Analyzer untuk mengakses CloudTrail data Anda atas nama Anda guna meninjau aktivitas akses guna mengidentifikasi layanan dan tindakan yang telah digunakan. Untuk mempelajari lebih lanjut tentang izin yang diperlukan untuk peran ini, lihat [izin diperlukan untuk menghasilkan kebijakan](#).
7. Di bagian CloudTrail jejak yang akan dianalisis, tentukan CloudTrail jejak yang mencatat peristiwa untuk akun tersebut.

Jika Anda memilih CloudTrail jejak yang menyimpan log di akun yang berbeda, kotak informasi tentang akses lintas akun akan ditampilkan. Akses lintas akun memerlukan 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 kebijakan menampilkan 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 bahwa 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 yang 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. Templat kebijakan berisi placeholder ARN sumber daya untuk tindakan yang mendukung izin tingkat sumber daya, seperti yang ditunjukkan pada gambar berikut. Izin tingkat sumber daya mengacu pada kemampuan untuk menentukan sumber daya mana yang boleh digunakan oleh para pengguna 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](#).

Generated policy

1 2 3

Customize permissions

Review the following policy template. You must specify resources for actions that support resource-level permissions to continue creating the policy.

The screenshot displays the AWS IAM console interface. On the left, a code editor shows a JSON policy template. The template includes two statements. The first statement allows actions like 'iam:ListUsers' and 'iam:ListRoles'. The second statement allows actions like 'iam:GetUser' and 'iam:ListUsers'. The resource for the second statement is set to 'arn:aws:iam::\${Account}:user/\${UserNameWithPath}'. On the right, the 'Edit statement' dialog box is open, showing a 'Select a statement' prompt and an 'Add new statement' button.

```

1 - {
2   "Version": "2012-10-17",
3   "Statement": [
4     {
5       "Effect": "Allow",
6       "Action": [
7         "access-analyzer:ValidatePolicy",
8         "iam:GetAccountPasswordPolicy",
9         "iam:GetAccountSummary",
10        "iam:ListAccountAliases",
11        "iam:ListGroups",
12        "iam:ListPolicies",
13        "iam:ListRoles",
14        "iam:ListUsers"
15      ],
16      "Resource": "*"
17    },
18    {
19      "Effect": "Allow",
20      "Action": [
21        "iam:GetRole",
22        "iam:ListAttachedRolePolicies",
23        "iam:ListInstanceProfilesForRole",
24        "iam:ListRolePolicies",
25        "iam:ListRoleTags"
26      ],
27      "Resource": "arn:aws:iam::${Account}:role/${RoleNameWithPath}"
28    },
29    {
30      "Effect": "Allow",
31      "Action": [
32        "iam:GetUser",
33        "iam:ListAccessKeys",
34        "iam:ListAttachedUserPolicies",
35        "iam:ListGroupsForUser",
36        "iam:ListUserTags"
37      ],
38      "Resource": "arn:aws:iam::${Account}:user/${UserNameWithPath}"
39    }
40  ]
41 }

```

2. (Opsional) Tambahkan, modifikasi, atau hapus pernyataan kebijakan JSON dalam templat. Untuk mempelajari selengkapnya 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 Lampirkan 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 [Menambahkan 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 yang mengatur. Misalnya, Anda dapat menggunakan AWS Organizations untuk membuat jejak yang mencatat semua peristiwa untuk semua yang ada Akun AWS di 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 melakukannya, Anda memerlukan peran dan kebijakan bucket yang memberikan izin

IAM Access Analyzer ke log Anda. CloudTrail Untuk informasi selengkapnya tentang membuat jalur Organizations, 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 ember di akun B. Sebelum Anda dapat membuat kebijakan, Anda harus membuat pembaruan berikut:

1. Pilih peran yang sudah ada, atau buat peran layanan baru yang memberikan akses IAM Access Analyzer ke bucket di akun B (tempat CloudTrail log Anda disimpan).
2. Verifikasi kebijakan kepemilikan objek bucket Amazon S3 dan 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 Buat kebijakan, opsi untuk Menggunakan peran yang ada telah 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 ini digunakan untuk memberikan IAM Access Analyzer akses ke CloudTrail log Anda di akun B.

Langkah 2: Verifikasi atau perbarui konfigurasi bucket Amazon S3 Anda di akun B

1. [Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/](https://console.aws.amazon.com/s3/).
2. Dalam daftar Bucket, pilih nama bucket tempat log CloudTrail jejak Anda disimpan.
3. Pilih tab Permissions dan pergi ke bagian Object Ownership.

Gunakan setelan bucket Kepemilikan Objek Amazon S3 untuk mengontrol kepemilikan objek yang Anda unggah ke bucket. Secara default, ketika objek Akun AWS upload lain ke bucket Anda, akun upload memiliki objek. Untuk membuat kebijakan, pemilik bucket harus memiliki semua objek di ember. Bergantung pada kasus penggunaan ACL, Anda mungkin perlu mengubah setelan Kepemilikan Objek untuk bucket Anda. Tetapkan Object Ownership ke salah satu opsi berikut.

- Pemilik bucket diberlakukan (disarankan)
- Pemilik ember lebih disukai

⚠ Important

Agar berhasil menghasilkan kebijakan, objek dalam ember harus dimiliki oleh pemilik ember. Jika Anda memilih untuk menggunakan pemilik Bucket yang diinginkan, Anda hanya dapat membuat kebijakan untuk jangka waktu tersebut 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 *DOC-EXAMPLE-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 Resource elemen Condition dalam memastikan bahwa peran hanya dapat mengakses aktivitas untuk akun jika itu milik akun A. Anda dapat mengganti *DOC-EXAMPLE-BUCKET* dengan nama bucket, 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:::DOC-EXAMPLE-BUCKET",
```

```

    "arn:aws:s3:::DOC-EXAMPLE-BUCKET/optional-prefix/AWSLogs/organization-id/
    ${aws:PrincipalAccount}/*"
  ],
  "Condition": {
    "StringEquals": {
      "aws:PrincipalOrgID": "organization-id"
    },
    "StringLike": {
      "aws:PrincipalArn": "arn:aws:iam::${aws:PrincipalAccount}:role/service-
      role/AccessAnalyzerMonitorServiceRole*"
    }
  }
}
]
}

```

5. Jika Anda mengenkripsi log Anda menggunakan AWS KMS, perbarui kebijakan AWS KMS kunci Anda di akun tempat Anda menyimpan CloudTrail log untuk memberikan akses kepada IAM Access Analyzer untuk menggunakan kunci Anda, seperti yang ditunjukkan dalam 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"
          ]
        }
      }
    }
  ]
}

```

```
        "aws:PrincipalArn": "arn:aws:iam::${aws:PrincipalAccount}:role/service-  
        role/AccessAnalyzerMonitorServiceRole*"  
    }  
  }  
} ]  
}
```

Menghasilkan kebijakan berdasarkan CloudTrail aktivitas (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](#)

Menghasilkan 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.

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

Layanan	Awalan layanan
AWS Profiler Biaya Aplikasi	application-cost-profiler
Wawasan CloudWatch Aplikasi Amazon	wawasan aplikasi
AWS App Mesh	appmesh
Amazon AppStream 2.0	appstream
AWS AppSync	appsync
Layanan Dikelola Amazon untuk Prometheus	aps
Amazon Athena	Athena
AWS Audit Manager	Manajer 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

Layanan	Awalan layanan
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
Kumpulan pengguna Amazon Cognito	cognito-idp
Sinkronisasi Amazon Cognito	sinkronis asi kognito
Amazon Comprehend Medical	memahamimedis
AWS Compute Optimizer	pengoptim al komputasi
AWS Config	config
Amazon Connect	hubungkan
AWS Cost and Usage Report	cur
AWS Glue DataBrew	databrew
AWS Data Exchange	pertukaran data

Layanan	Awalan layanan
AWS Data Pipeline	datapipeline
Akselerator DynamoDB	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
AWS Directory Service	ds
Amazon DynamoDB	dynamodb
Toko Blok Elastis Amazon	ebs
Awan Komputasi Elastis Amazon	ec2
Amazon Elastic Container Registry	ecr
Amazon Elastic Container Registry Publik	ecr-publik
Layanan Kontainer Elastis Amazon	ecs
Layanan Amazon Elastic Kubernetes	eks
Amazon Elastic Inference	inferensi elastis
Amazon ElastiCache	elasticache
AWS Elastic Beanstalk	elasticbeanstalk
Amazon Elastic File System	sistem file elastis

Layanan	Awalan layanan
Elastic Load Balancing	elastico adbalancing
Amazon Elastic Transcoder	elastictranscoder
Amazon EMR di EKS (Wadah EMR)	kontainer emr
Amazon EMR Tanpa Server	emr-tanpa server
OpenSearch Layanan Amazon	es
Amazon EventBridge	peristiwa
Amazon CloudWatch Terbukti	ternyata
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
Amazon GameLift	gamelift
Amazon Location Service	geo
Gletser Amazon S3	gletser
Grafana yang Dikelola Amazon	grafana
AWS IoT Greengrass	greengrass
AWS Ground Station	stasiun tanah
Amazon GuardDuty	tugas jaga

Layanan	Awalan layanan
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
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

Layanan	Awalan layanan
Amazon Kinesis	kinesis
Amazon Kinesis Analytics V2	kinesisanalitik
AWS Key Management Service	km
AWS Lambda	lambda
Amazon Lex	lex
AWS License Manager Manajer Langganan Linux	license-manager- linux-subscriptions
Amazon Lightsail	lightsail
CloudWatch Log Amazon	log
Amazon Lookout for Equipment	peralatan pencarian
Amazon Lookout for Metrics	lookoutmetrics
Amazon Lookout for Vision	lookoutvision
AWS Mainframe Modernization	m2
Blockchain yang Dikelola Amazon	terkelolablockchain
AWS Elemental MediaConnect	mediaconnect
AWS Elemental MediaConvert	mediaconvert
AWS Elemental MediaLive	medialive
AWS Elemental MediaPackage	mediapackage
AWS Elemental MediaPackage VOD	mediapackage-vod
AWS Elemental MediaStore	mediastore

Layanan	Awalan layanan
AWS Elemental MediaTailor	mediatailor
Amazon MemoryDB for Redis	memorydb
AWS Application Migration Service	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 Panorama	panorama
AWS Wawasan Kinerja	pi
EventBridge Pipa Amazon	pipa
Amazon Polly	polly
Profil Pelanggan Amazon Connect	profile
Amazon QLDB	qldb

Layanan	Awalan layanan
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
AWS Resource Groups	kelompok sumber daya
AWS RoboMaker	pembuat robomak
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-pemulihan-kesiapan
Amazon Route 53 Resolver	route53resolver
AWS CloudWatch RUM	rum

Layanan	Awalan layanan
Layanan Penyimpanan Sederhana Amazon	s3
Amazon S3 on Outposts	pos terdepan s3
Kemampuan SageMaker geospasial Amazon	pembuat sagemer- geospasial
Savings Plans	savingsplans
EventBridge Skema Amazon	skema
Amazon SimpleDB	sdb
AWS Secrets Manager	manajer rahasia
AWS Security Hub	securityhub
Danau Keamanan Amazon	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	penandatanganan
AWS SimSpace Weaver	simspaceweaver
AWS Server Migration Service	sms
Amazon Pinpoint SMS dan Layanan Suara	sms-suara

Layanan	Awalan layanan
AWS Snowball	bola salju
Layanan Antrian Sederhana Amazon	sq3
AWS Systems Manager	ssm
AWS Systems Manager Incident 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	texttract
Amazon Timestream	aliran waktu
AWS Pembangun Jaringan Telekomunikasi	tnb
Amazon Transcribe	mentranskripsikan
AWS Transfer Family	transfer
Amazon Translate	menerjemahkan
ID Suara Amazon Connect	voiceid
Kisi VPC Amazon	vpc-kisi
AWS WAFV2	wafv2
AWS Well-Architected Tool	wellarchitected

Layanan	Awalan layanan
Kebijakan Amazon Connect	kebijaksanaan
Amazon WorkLink	tautan kerja
Amazon WorkSpaces	ruang kerja
AWS X-Ray	xray

Kuota Penganalisis Akses IAM

IAM Access Analyzer memiliki kuota berikut:

Sumber daya	Kuota default	Kuota maksimum
Penganalisis tingkat akun maksimum per jenis penganalisis per Wilayah Akun AWS	1	1
Penganalisis tingkat organisasi maksimum per jenis penganalisis per Wilayah Akun AWS	5	20 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

Sumber daya	Kuota default	Kuota maksimum
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
Pembuatan kebijakan per hari	Afrika (Cape Town): 5 Asia Pasifik (Hong Kong): 5 Eropa (Milan): 5 Timur Tengah (Bahrain): 5 Semua wilayah lain yang didukung: 50	Afrika (Cape Town): 5 Asia Pasifik (Hong Kong): 5 Eropa (Milan): 5 Timur Tengah (Bahrain): 5 Semua wilayah lain yang didukung: 50

 **Note**
 Permintaan pembuatan kebijakan yang dibatalkan berlaku untuk kuota harian.

[Beberapa kuota dapat dikonfigurasi pelanggan menggunakan Service Quotas.](#)

Menyelesaikan masalah IAM

Jika Anda mengalami masalah tentang akses ditolak atau kesulitan serupa saat bekerja dengan AWS Identity and Access Management (IAM), periksa topik di bagian ini.

Topik

- [Memecahkan masalah umum IAM](#)
- [Memecahkan masalah akses ditolak pesan kesalahan](#)
- [Memecahkan masalah kebijakan IAM](#)
- [Mengonfigurasi tombol keamanan FIDO](#)
- [Memecahkan masalah peran IAM](#)
- [Memecahkan Masalah IAM dan Amazon EC2](#)
- [Memecahkan Masalah IAM dan Amazon S3](#)
- [Memecahkan masalah federasi SAML 2.0 dengan AWS](#)

Memecahkan masalah umum IAM

Gunakan informasi di sini untuk membantu Anda mendiagnosis dan memperbaiki masalah umum saat Anda bekerja sama AWS Identity and Access Management (IAM).

Masalah

- [Saya tidak dapat masuk ke akun AWS saya](#)
- [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](#)

Saya tidak dapat masuk ke akun AWS saya

Verifikasi bahwa Anda memiliki kredensial yang benar dan bahwa Anda menggunakan metode yang benar untuk masuk. Untuk informasi lebih lanjut, lihat [Memecahkan masalah login](#) di dalam AWS Sign-In Panduan 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.

Untuk informasi selengkapnya, lihat [Mengatur ulang kata sandi yang hilang atau terlupakan atau access key untuk AWS](#).

Variabel kebijakan tidak berfungsi

- 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. Setiap kebijakan yang tidak mencakup variabel akan tetap berlaku jika Anda memasukkan nomor versi terbaru.

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 IAM JSON:Version](#). Untuk mempelajari selengkapnya tentang versi kebijakan, lihat [the section called "Peningkatan versi IAM"](#).

- Verifikasi bahwa variabel kebijakan Anda berada di kasus yang tepat. Untuk detailnya, lihat [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 lainnya AWS layanan), termasuk tag yang digunakan dalam [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, dari zona replikasi ke zona replikasi, dan dari Wilayah ke Wilayah di seluruh dunia. 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 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 tentang bagaimana beberapa layanan AWS lainnya dipengaruhi oleh hal ini, pelajari sumber daya berikut:

- Amazon DynamoDB: [Apa model konsistensi Amazon DynamoDB?](#) di dalam FAQ DynamoDB, dan [Baca Konsistensi](#) di Panduan Pengembang Amazon DynamoDB.
- Amazon EC2: [Konsistensi Akhirnya EC2](#) di dalam Referensi API Amazon EC2.
- Amazon ESDM: [Memastikan Konsistensi Saat Menggunakan Amazon S3 dan Amazon ElasticMapReduce untuk Alur Kerja ETL](#) di dalam AWS Blog Data Besar
- Pergeseran Merah Amazon: [Mengelola Konsistensi Data](#) di dalam Panduan Pengembang Database Amazon Redshift
- Amazon: [Model Konsistensi Data Amazon S3](#) di 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 menciptakan perangkat MFA virtual untuk pengguna di IAM tetapi tidak pernah menetapkannya kepada pengguna. Anda harus menghapus perangkat MFA virtual yang ada sebelum Anda dapat 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 CLI atau AWS API untuk menghapus perangkat MFA virtual yang ada namun belum ditetapkan.

Untuk menghapus perangkat MFA virtual yang ada namun belum ditetapkan

1. Lihat perangkat MFA virtual di akun Anda.
 - AWS CLI: [aws iam list-virtual-mfa-devices](#)
 - AWSAPI: [ListVirtualMFADevices](#)
2. Sebagai 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](#)
 - AWSAPI: [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 [gunakan IAM Identity Center untuk otentikasi](#). Jika Anda menggunakan IAM, AWS merekomendasikan agar 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 berikut untuk membuat pengguna baru dengan aman di IAM:

1. [Buat pengguna baru](#) menggunakan AWS Management Console. Pilih untuk memberi AWS Management Console akses dengan kata sandi yang dibuat secara otomatis. 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 tautan konsol pengguna IAM dan nama pengguna mereka kepada pengguna Anda. 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 dibuat AWS dan menyimpan informasi itu 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 [Mencatat panggilan IAM dan AWS STS API dengan AWS CloudTrail](#).
- [AWS Pusat Pengetahuan](#)- 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 akses ditolak pesan kesalahan

Kesalahan akses ditolak muncul ketika AWS secara eksplisit atau implisit menolak permintaan otorisasi. Penolakan eksplisit terjadi ketika kebijakan berisi Deny pernyataan untuk tindakan tertentu AWS. Penolakan implisit terjadi ketika tidak ada pernyataan yang berlaku dan juga tidak ada Deny pernyataan yang berlaku. Karena kebijakan IAM menyangkal prinsip IAM secara default, kebijakan tersebut harus secara eksplisit mengizinkan prinsipal untuk melakukan suatu tindakan. Jika tidak, kebijakan tersebut secara implisit menolak akses. Untuk informasi selengkapnya, lihat [Perbedaan antara penolakan tegas dan implisit](#).

Jika beberapa kebijakan dari jenis kebijakan yang sama menolak permintaan otorisasi, maka AWS tidak menentukan jumlah kebijakan dalam pesan kesalahan akses ditolak. Jika beberapa jenis kebijakan menolak permintaan otorisasi, hanya AWS menyertakan salah satu jenis kebijakan tersebut dalam pesan kesalahan.

Important

Mengalami masalah saat 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 kredensial yang Anda atur saat membuat Akun AWS. Jika Anda adalah pengguna IAM, administrator akun Anda dapat memberi Anda kredensial yang dapat Anda gunakan untuk masuk. AWS Jika Anda perlu meminta dukungan, jangan gunakan tautan umpan balik di halaman ini, karena formulir diterima oleh tim AWS Dokumentasi, bukan AWS Support. Sebagai gantinya, pada halaman [Hubungi 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 kebijakan yang bertanggung jawab untuk menolak akses. Misalnya, jika kesalahan menyebutkan bahwa akses ditolak karena Kebijakan Kontrol Layanan (SCP), maka Anda dapat fokus pada pemecahan masalah SCP. Bila Anda mengetahui jenis kebijakan, Anda juga dapat memeriksa pernyataan penolakan atau izin yang hilang pada tindakan spesifik dalam kebijakan jenis kebijakan tersebut. Jika pesan kesalahan tidak

menyebutkan jenis kebijakan yang bertanggung jawab untuk menolak akses, gunakan pedoman 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 [Mengelola kebijakan IAM](#).
- Jika AWS Management Console mengembalikan pesan yang menyatakan bahwa Anda tidak berwenang untuk melakukan tindakan, maka Anda harus menghubungi administrator Anda untuk 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 izin `widgets:GetWidget` 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, verifikasi 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 hanya berbeda menurut kasus, akses

Anda mungkin akan ditolak secara tak terduga. Untuk informasi selengkapnya, lihat [Elemen kebijakan IAM JSON: Condition](#).

- Jika Anda memiliki [batas izin](#), verifikasi 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 penanggung jawab IAM (pengguna atau peran). Kebijakan berbasis sumber daya tidak terbatas oleh batasan izin. Batas izin bukanlah hal yang umum. Untuk informasi selengkapnya tentang cara AWS mengevaluasi kebijakan, lihat [Logika evaluasi kebijakan](#).
- Jika Anda menandatangani permintaan secara manual (tanpa menggunakan [SDK AWS](#)), verifikasi bahwa Anda sudah [menandatangani permintaan](#) dengan benar.

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 [Menggunakan kredensial sementara dengan sumber daya AWS](#). 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 semmentaranya telah Anda gunakan. Untuk informasi selengkapnya tentang bagaimana izin untuk kredensial keamanan sementara ditentukan, lihat [Mengontrol izin untuk kredensial keamanan sementara](#).
- Jika Anda mengambil peran, sesi peran Anda mungkin dibatasi oleh kebijakan sesi. [Bila Anda meminta kredensial keamanan sementara secara terprogram menggunakan AWS STS, Anda dapat secara opsional meneruskan kebijakan sesi inline atau terkelola](#). 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 pengguna gabungan, sesi Anda mungkin akan dibatasi oleh kebijakan sesi. Anda menjadi pengguna federasi dengan masuk AWS sebagai pengguna IAM dan kemudian meminta token federasi. Untuk informasi selengkapnya tentang pengguna gabungan, lihat [GetFederationToken—federasi melalui pialang 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 `MyBucket`.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "S3BucketPolicy",
    "Effect": "Allow",
    "Principal": {"AWS": ["arn:aws:iam::111122223333:role/MyRole"]},
    "Action": ["s3:PutObject"],
    "Resource": ["arn:aws:s3:::MyBucket/*"]
  }]
}
```

Akses ditolak contoh pesan kesalahan

Sebagian besar pesan kesalahan akses ditolak muncul dalam formatUser *user* is not authorized to perform *action* on *resource* because *context*. Dalam contoh ini, *pengguna* adalah [Nama Sumber Daya Amazon \(ARN\)](#) yang tidak menerima akses, *tindakan* adalah tindakan layanan yang ditolak kebijakan, dan *sumber daya* adalah ARN sumber daya tempat kebijakan bertindak. *KoLom* konteks mewakili konteks tambahan tentang jenis kebijakan yang menjelaskan mengapa kebijakan menolak akses.

Ketika kebijakan secara eksplisit menolak akses karena kebijakan berisi Deny pernyataan, maka AWS sertakan frasa `with an explicit deny in a type policy` dalam pesan kesalahan akses ditolak. Ketika kebijakan secara implisit menolak akses, maka AWS sertakan frasa `because no type policy allows the action` action dalam pesan kesalahan akses ditolak.

Note

Beberapa AWS layanan tidak mendukung akses ini ditolak format pesan kesalahan. 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 adalah `codecommit:ListRepositories`.
2. Perbarui SCP Anda dengan menambahkan Allow pernyataan. Untuk informasi selengkapnya, lihat [Memperbarui SCP](#) di Panduan AWS Organizations Pengguna.

```
User: arn:aws:iam::777788889999:user/JohnDoe is not authorized to perform:
codecommit:ListRepositories because no service control policy allows the
codecommit:ListRespositories action
```

Akses ditolak karena Kebijakan Kontrol Layanan — penolakan eksplisit

1. Periksa Deny pernyataan untuk tindakan dalam Kebijakan Kontrol Layanan (SCP) Anda. Untuk contoh berikut, tindakannya adalah `codecommit:ListRepositories`.
2. Perbarui SCP Anda dengan menghapus Deny pernyataan. Untuk informasi selengkapnya, lihat [Memperbarui SCP](#) di Panduan AWS Organizations Pengguna.

```
User: arn:aws:iam::777788889999:user/JohnDoe is not authorized to perform:
codecommit:ListRepositories with an explicit deny in a service 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 pernyataan. Allow Untuk informasi selengkapnya, lihat [Memperbarui kebijakan titik akhir VPC di Panduan](#).AWS PrivateLink

```
User: arn:aws:iam::123456789012:user/JohnDoe 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 pernyataan. Deny Untuk informasi selengkapnya, lihat [Memperbarui kebijakan titik akhir VPC di Panduan](#).AWS PrivateLink

```
User: arn:aws:iam::123456789012:user/JohnDoe 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. Lihat informasi yang lebih lengkap di [Batas izin untuk entitas IAM](#) dan [Menyunting Kebijakan IAM](#).

```
User: arn:aws:iam::123456789012:user/JohnDoe 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. Periksa Deny pernyataan eksplisit untuk tindakan di batas izin Anda. Untuk contoh berikut, tindakannya adalah `sagemaker:ListModelIs`.
2. Perbarui batas izin Anda dengan menghapus Deny pernyataan dari kebijakan IAM Anda. Lihat informasi yang lebih lengkap di [Batas izin untuk entitas IAM](#) dan [Menyunting Kebijakan IAM](#).

```
User: arn:aws:iam::777788889999:user/JohnDoe is not authorized to perform:
sagemaker:ListModelIs 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](#) dan [Menyunting Kebijakan IAM](#).

```
User: arn:aws:iam::123456789012:user/JohnDoe is not authorized to perform:
codecommit:ListRepositories because no session policy allows the
codecommit:ListRepositories action
```

Akses ditolak karena kebijakan sesi — penolakan eksplisit

1. Periksa Deny pernyataan eksplisit untuk tindakan dalam kebijakan sesi Anda. Untuk contoh berikut, tindakannya adalah `codedeploy:ListDeployments`.
2. Perbarui kebijakan sesi Anda dengan menghapus Deny pernyataan. Untuk informasi selengkapnya, lihat [Kebijakan sesi](#) dan [Menyunting Kebijakan IAM](#).

```
User: arn:aws:iam::123456789012:user/JohnDoe is not authorized to perform:
codedeploy:ListDeployments on resource: arn:aws:codedeploy:us-
east-1:123456789012:deploymentgroup:* with an explicit deny in a sessions 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 selengkapnya, lihat [Kebijakan berbasis sumber daya](#) dan [Menyunting Kebijakan IAM](#)

```
User: arn:aws:iam::123456789012:user/JohnDoe 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 selengkapnya, lihat [Kebijakan berbasis sumber daya](#) dan [Menyunting Kebijakan IAM](#)

```
User: arn:aws:iam::123456789012:user/JohnDoe 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 selengkapnya, lihat [Kebijakan berbasis sumber daya](#) dan [Menyunting Kebijakan IAM](#)

```
User: arn:aws:iam::123456789012:user/JohnDoe 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 selengkapnya, lihat [Kebijakan berbasis sumber daya](#) dan [Menyunting Kebijakan IAM](#)

```
User: arn:aws:iam::777788889999:user/JohnDoe 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 `codecommit:ListRepositories` dilampirkan ke pengguna `JohnDoe`.
2. Perbarui kebijakan Anda dengan menambahkan Allow pernyataan. Untuk informasi selengkapnya, lihat [Kebijakan berbasis identitas](#) dan [Menyunting Kebijakan IAM](#)

```
User: arn:aws:iam::123456789012:user/JohnDoe 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. Periksa Deny pernyataan eksplisit untuk tindakan dalam kebijakan berbasis identitas yang dilampirkan pada identitas. Untuk contoh berikut, tindakan `codedeploy:ListDeployments` dilampirkan ke pengguna `JohnDoe`.
2. Perbarui kebijakan Anda dengan menghapus Deny pernyataan. Untuk informasi selengkapnya, lihat [Kebijakan berbasis identitas](#) dan [Menyunting Kebijakan IAM](#)

```
User: arn:aws:iam::123456789012:user/JohnDoe 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
```


Akses ditolak ketika permintaan VPC gagal karena kebijakan lain

1. Periksa Deny pernyataan eksplisit untuk tindakan dalam Kebijakan Kontrol Layanan (SCP) Anda. Untuk contoh berikut, tindakannya adalah `SNS:Publish`.
2. Perbarui SCP Anda dengan menghapus Deny pernyataan. Untuk informasi selengkapnya, lihat [Memperbarui SCP](#) di Panduan AWS IAM Identity Center Pengguna.

```
User: arn:aws:sts::111122223333:assumed-role/role-name/role-session-name is not
authorized to perform:
SNS:Publish on resource: arn:aws:sns:us-east-1:444455556666:role-name-2
with an explicit deny in a VPC endpoint policy transitively through a service control
policy
```

Memecahkan masalah kebijakan IAM

[Kebijakan](#) adalah entitas dalam AWS yang, ketika dilampirkan ke identitas atau sumber daya, menetapkan izinnya. AWS mengevaluasi kebijakan ini ketika prinsipal, seperti pengguna, membuat permintaan. Izin dalam kebijakan menentukan apakah permintaan tersebut diizinkan atau ditolak. Kebijakan disimpan di AWS sebagai dokumen JSON yang dilampirkan ke prinsipal sebagai kebijakan berbasis identitas atau ke sumber daya sebagai kebijakan berbasis sumber daya. Anda dapat melampirkan kebijakan berbasis identitas pada penanggung jawab (atau identitas) seperti grup IAM, pengguna, atau peran. Kebijakan berbasis identitas dapat mencakup kebijakan terkelola AWS, kebijakan terkelola pelanggan, dan kebijakan inline. Anda dapat membuat dan mengedit kebijakan yang dikelola pelanggan AWS Management Console dengan menggunakan opsi editor Visual dan JSON. Saat Anda melihat kebijakan di AWS Management Console, 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 AWS Management Console. Untuk mempelajari lebih lanjut tentang sintaksis JSON di kebijakan IAM, lihat [Tata bahasa kebijakan IAM JSON](#).

Memecahkan 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](#)
 - [Perbaiki layanan, tindakan, atau tipe sumber daya yang tidak dikenal di editor visual](#)
- [Pemecahan masalah menggunakan 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 Anda membuat atau mengedit kebijakan yang dikelola pelanggan, Anda dapat menggunakan informasi di Editor visual untuk membantu Anda memecahkan kesalahan di 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. Ketika AWS mengembalikan kebijakan sebagai respons atas kueri pengguna atau menampilkannya di konsol, AWS mengubah kebijakan kembali menjadi format yang dapat dibaca manusia tanpa mengubah izin yang diberikan oleh kebijakan. Hal ini dapat mengakibatkan perbedaan dalam hal 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 di dalam elemen prinsipal dapat diganti dengan ARN 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 AWS Management Console, Anda dapat memilih untuk bekerja sepenuhnya di editor JSON. Jika Anda tidak pernah melakukan perubahan di Editor visual dan memilih Berikutnya dari editor JSON, lebih kecil kemungkinannya bagi kebijakan untuk direstrukturisasi. Namun, jika Anda membuat kebijakan dan menggunakan editor visual untuk melakukan modifikasi, atau jika Anda memilih Berikutnya dari opsi Editor visual, maka IAM mungkin merestrukturisasi kebijakan untuk mengoptimalkan tampilannya di editor visual.

Restrukturisasi ini hanya ada pada sesi pengeditan Anda dan tidak disimpan secara otomatis.

Jika kebijakan Anda distruktur ulang di sesi pengeditan, IAM menentukan apakah akan menyimpan restrukturisasi berdasarkan situasi berikut:

Menggunakan opsi editor ini	Jika Anda mengedit kebijakan	Dan kemudian pilih Berikutnya 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

Menggunakan opsi editor ini	Jika Anda mengedit kebijakan	Dan kemudian pilih Berikutnya dari tab ini	Saat Anda memilih Simpan perubahan
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 jenis sumber daya.

- Gunakan pembangun ARN – Berdasarkan jenis sumber daya, Anda mungkin melihat bidang yang berbeda untuk membangun ARN Anda. 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`. Anda harus memberikan nilai Wilayah, Akun, dan InstanceId untuk sumber daya Anda. Jika Anda memberikan ID akun tetapi memilih Apa pun untuk Wilayah dan ID instans, maka kebijakan tersebut memberikan izin untuk setiap instans dalam akun Anda.
- 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: Resource](#).

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

ditolak secara default, kami merekomendasikan sebagai praktik terbaik agar Anda memberikan izin hanya untuk tindakan dan sumber daya yang dibutuhkan pengguna. Anda harus membuat pernyataan untuk menolak izin hanya jika Anda ingin membatalkan izin secara terpisah yang diizinkan oleh pernyataan atau kebijakan lain. Kami sarankan Anda membatasi jumlah izin penolakan seminim mungkin karena dapat meningkatkan kesulitan izin pemecahan masalah. Untuk informasi lebih lanjut tentang cara IAM mengevaluasi logika kebijakan, lihat [Logika evaluasi kebijakan](#).

Note

Secara default, hanya Pengguna root akun AWS yang memiliki akses ke semua sumber daya dalam akun tersebut. 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 mengizinkan 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.

Jika Anda akrab dengan sintaksis JSON, Anda juga dapat menggunakan karakter wildcard (*) untuk menetapkan 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 sintaksis JSON (seperti wildcard) untuk layanan, membuat, mengedit, dan menyimpan 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 dan tidak mengizinkan Anda untuk meninjau dan menyimpan kebijakan Anda. 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 kesalahan mencakup jumlah karakter yang dimuat dalam dokumen kebijakan Anda, dan Anda dapat menggunakan informasi ini untuk membantu Anda mengurangi ukuran kebijakan Anda.

Perbaiki layanan, tindakan, atau tipe sumber daya yang tidak dikenal di editor visual

Saat Anda membuat atau mengedit kebijakan dalam editor visual, Anda mungkin melihat peringatan bahwa kebijakan Anda mencakup layanan, tindakan, atau jenis sumber daya yang tidak dikenal.

 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 di pratinjau, Anda dapat mengabaikan peringatan dan melanjutkan, meskipun Anda harus secara manual mengetikkan ARN tindakan dan sumber daya untuk melengkapi kebijakan Anda. Atau, Anda dapat memilih opsi editor JSON untuk mengetik atau menempel dokumen kebijakan JSON.
- Layanan khusus – Layanan khusus tidak mendukung editor visual. Jika Anda menggunakan layanan khusus, Anda dapat mengabaikan peringatan dan melanjutkan, meskipun Anda harus secara manual mengetikkan ARN tindakan dan sumber daya untuk melengkapi kebijakan Anda. Atau, Anda dapat memilih opsi editor JSON untuk mengetik atau menempel dokumen kebijakan JSON.
- Layanan tidak mendukung editor visual — Jika kebijakan Anda mencakup layanan yang tersedia secara umum (GA) yang tidak mendukung editor visual, Anda dapat mengabaikan peringatan dan melanjutkan, meskipun Anda harus mengetik tindakan dan sumber daya ARNs secara manual untuk melengkapi kebijakan Anda. Atau, Anda dapat memilih opsi editor JSON untuk mengetik atau menempel 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 mencakup layanan terdukung dengan tindakan yang tidak didukung, Anda dapat mengabaikan peringatan dan melanjutkan, meskipun Anda harus menyetikkan ARN sumber daya secara manual untuk melengkapi kebijakan Anda. Atau, Anda dapat memilih opsi editor JSON untuk mengetik atau menempel 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 menyetikkan layanan, tindakan, atau sumber daya secara manual dalam editor visual, Anda dapat membuat kebijakan yang mencakup kesalahan ketik. Untuk menghindari hal ini, sebaiknya gunakan editor visual dengan memilih dari daftar layanan dan tindakan, kemudian lengkapi bagian sumber daya sesuai dengan petunjuk. Namun demikian, jika layanan tidak sepenuhnya mendukung editor visual, Anda mungkin harus menyetikkan 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 apakah ada nama layanan, tindakan, dan tipe sumber daya salah eja. Misalnya, Anda mungkin menggunakan `s2` alih-alih `s3` dan `ListMyBuckets` alih-alih `ListAllMyBuckets`. Salah satu tindakan umum lainnya adalah masuknya teks yang tidak perlu di ARN, seperti `arn:aws:s3:::*`, atau titik dua hilang dalam tindakan, seperti `iam.CreateUser`. Anda dapat mengevaluasi kebijakan yang mungkin menyertakan kesalahan ketik dengan memilih Berikutnya untuk meninjau ringkasan kebijakan dan mengonfirmasi apakah kebijakan memberikan izin yang Anda inginkan.

Pemecahan masalah menggunakan 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. Anda dapat melihat [ringkasan kebijakan](#) untuk setiap kebijakan untuk setiap kebijakan untuk setiap kebijakan untuk kebijakan tersebut. Anda dapat melihat ringkasan kebijakan untuk kebijakan yang dikelola pada halaman Kebijakan. Jika AWS tidak dapat membuat ringkasan kebijakan, Anda melihat dokumen kebijakan JSON dan bukan ringkasan, serta menerima 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

Anda harus berhati-hati saat menggunakan elemen-elemen kebijakan ini seperti `NotPrincipal`, `NotAction`, dan `NotResource`. Untuk informasi selengkapnya tentang elemen-elemen kebijakan, lihat [Referensi elemen kebijakan IAM JSON](#).

Anda dapat membuat kebijakan yang tidak memberikan izin yang efektif jika Anda memberikan layanan dan sumber daya yang tidak sesuai. Hal ini dapat terjadi jika 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. Untuk lebih memahami kebijakan Anda, selalu uji kebijakan tersebut dengan [simulator kebijakan](#).

Ringkasan kebijakan mencakup layanan, tindakan, atau tipe sumber daya yang tidak dikenal

Di konsol IAM, jika [ringkasan kebijakan](#) menyertakan simbol peringatan

() ,

kebijakan tersebut mungkin mencakup layanan, tindakan, atau tipe sumber daya yang tidak dikenal. 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 tik —AWS memeriksa JSON benar secara sintaksis, dan kebijakan tidak menyertakan salah tik atau kesalahan lainnya sebagai bagian dari [validasi kebijakan](#).

Note

Sebagai [praktik terbaik](#), kami menyarankan Anda menggunakan IAM Access Analyzer untuk memvalidasi kebijakan IAM Anda untuk memastikan izin yang aman dan fungsional. Sebaiknya buka kebijakan yang ada dan tinjau serta selesaikan rekomendasi validasi kebijakan.

Layanan tidak mendukung ringkasan kebijakan IAM

Ketika layanan atau tindakan yang tersedia secara umum (GA) tidak dikenali oleh ringkasan kebijakan IAM atau editor visual, kemungkinan layanan tersebut tidak mendukung fitur-fitur ini. 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 AWS Management Console 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 footer AWS Management Console, pilih Umpan Balik. Di kotak Umpan 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 AWS Management Console 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 footer AWS Management Console, 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 Layanan AWS](#).

Untuk mempelajari apakah kebijakan Anda mendefinisikan tindakan, sumber daya, atau kondisi yang tidak memberikan izin, Anda dapat melihat [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](#)
- [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 satu pun tindakan `ec2:Describe` diberikan karena semuanya tidak mendukung izin tingkat sumber daya. Izin tingkat sumber daya berarti tindakan tersebut mendukung sumber daya yang menggunakan [ARN](#) di elemen [Resource](#) 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](#).

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": "ec2:Describe*",
    "Resource": "arn:aws:ec2:us-east-2:ACCOUNT-ID: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`.

```
{
  "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.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": "cloudfront:*",
    "Resource": [
      "arn:aws:cloudfront:*",
      "arn:aws:s3:::examplebucket"
    ]
  }]
}
```

Kebijakan ini memberikan izin untuk semua CloudFront tindakan. Tetapi karena kebijakan mendefinisikan sumber daya examplebucket 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.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": [
      "cloudfront:*",
      "s3:CreateBucket",
      "s3:ListBucket*",
      "s3:PutBucket*",
      "s3:GetBucket*"
    ],
    "Resource": [
      "arn:aws:cloudfront:*",
      "arn:aws:s3:::examplebucket"
    ]
  }]
}
```

Atau, untuk memperbaiki kebijakan ini agar hanya menyediakan 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 prefiks S3 samacustom dan ID versi sama dengan ID versi1234. 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 Layanan AWS](#) dan ikuti tautan ke dokumentasi layanan untuk kunci kondisi.

```
{
  "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`.

```
{
  "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`.

```
{
```



```
"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 kebijakan yang dikelola AWS ditautkan dengan layanan. Kebijakan ini hanya digunakan dengan [peran yang berkaitan dengan layanan](#) untuk layanan itu. Di konsol IAM, saat Anda melihat halaman Detail kebijakan untuk kebijakan untuk kebijakan, laman tersebut memuat banner untuk menunjukkan kebijakan tersebut ditautkan dengan 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 melakukannya, lihat peristiwa akun Anda di Riwayat CloudTrail peristiwa. CloudTrail log peristiwa mencakup informasi peristiwa terperinci yang dapat Anda gunakan untuk mengubah izin kebijakan. Anda mungkin menemukan bahwa pengguna atau peran mencoba melakukan tindakan di AWS dan permintaan tersebut ditolak. Dalam hal ini, Anda dapat mempertimbangkan apakah pengguna atau peran harus memiliki izin untuk melakukan tindakan tersebut. Jika demikian, Anda dapat menambahkan tindakan dan bahkan ARN dari sumber daya yang mereka coba akses ke kebijakan mereka. Sebagai alternatif, jika pengguna atau peran memiliki izin yang tidak mereka gunakan, Anda mungkin mempertimbangkan 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 [Melihat CloudTrail Peristiwa di CloudTrail Konsol](#) di PanduanAWS 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 sintaksis JSON, sementara IAM Access Analyzer memberikan pemeriksaan kebijakan tambahan dengan rekomendasi untuk membantu Anda lebih menyempurnakan kebijakan Anda. Untuk mempelajari selengkapnya tentang validasi kebijakan, lihat [Memvalidasi 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

DiAWS Management Console, Anda mungkin menerima kesalahan berikut jika Anda tidak memiliki izin untuk melihat hasil validasi kebijakan 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 satu dan hanya satu objek JSON. Anda menunjukkan sebuah objek dengan menempatkan runtkup { } di sekitarnya. Meskipun Anda dapat membuat nest untuk objek lain di dalam objek JSON dengan menyematkan runtkup { } tambahan di dalam pasangan bagian luar, kebijakan hanya dapat berisi satu pasang runtkup { } terluar. Contoh berikut ini salah karena berisi dua objek di tingkat atas (ditampilkan dalam *warna merah*):

```
{
  "Version": "2012-10-17",
  "Statement":
  {
    "Effect": "Allow",
    "Action": "ec2:Describe*",
    "Resource": "*"
  }
}
{
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:*",
    "Resource": "arn:aws:s3:::my-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):

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "ec2:Describe*",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
```

```
    "Action": "s3:*",
    "Resource": "arn:aws:s3:::my-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 runtkup { } 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 runtkup { }, yang berisi satu elemen Effect, satu elemen Action, dan satu elemen Resource. Contoh berikut ini salah karena berisi dua elemen Statement dalam objek kebijakan (ditampilkan dalam *warna merah*):

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "ec2:Describe*",
    "Resource": "*"
  },
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:*",
    "Resource": "arn:aws:s3:::my-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):

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

```
    "Action": "ec2:Describe*",
    "Resource": "*"
  },
  {
    "Effect": "Allow",
    "Action": "s3:* ",
    "Resource": "arn:aws:s3:::my-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

Di sisi nilai pasangan nama/nilai `Statement`, objek harus terdiri dari hanya satu elemen `Effect`, satu elemen `Action`, dan satu elemen `Resource`. Kebijakan berikut ini salah karena memiliki dua elemen `Effect` dalam objek nilai `Statement`:

```
{
  "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 elemen Action: semua elemen Action digabung secara internal dan diperlakukan seolah-olah itu adalah daftar tunggal.
- Beberapa elemen Resource: semua elemen Resource digabung secara internal dan diperlakukan seolah-olah itu adalah daftar tunggal.

Mesin kebijakan tidak memungkinkan Anda untuk menyimpan kebijakan dengan kesalahan sintaksis. Anda harus memperbaiki kesalahan di kebijakan sebelum dapat menyimpannya. Sebaiknya tinjau 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:

```
{
  "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 elemen Resource (yang ditampilkan dalam *warna merah*):

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:*",
    "Resource": "arn:aws:s3::my-bucket",
    "Resource": "arn:aws:s3::my-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):

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:*",
    "Resource": [
      "arn:aws:s3::my-bucket",
      "arn:aws:s3::my-bucket/*"
    ]
  }
}
```

Elemen versi JSON hilang

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 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 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. Namun, kebijakan ini gagal menentukan versi sintaksis kebijakan yang mendukung variabel kebijakan (ditampilkan dalam *warna merah*):

```
{
```

```
"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):

```
{
  "Version": "2012-10-17",
  "Statement":
  {
    "Action": "iam:*AccessKey*",
    "Effect": "Allow",
    "Resource": "arn:aws:iam::123456789012:user/${aws:username}"
  }
}
```

Mengonfigurasi tombol 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 dapat masuk menggunakan kunci keamanan FIDO saya](#)
- [Saya menghilangkan 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 Anda, periksa hal berikut:

- Apakah Anda menggunakan konfigurasi yang didukung?

Untuk informasi tentang perangkat dan browser yang dapat Anda gunakan dengan WebAuthn dan AWS, lihat [Konfigurasi yang didukung untuk menggunakan kunci keamanan FIDO](#).

- Apakah Anda menggunakan Mozilla Firefox?

Versi Firefox saat ini mendukung secara WebAuthn default. Untuk mengaktifkan dukungan WebAuthn di Firefox, lakukan hal berikut:

1. Dari bilah alamat Firefox, ketik **about:config**.
2. Pada bilah Pencarian dari layar yang terbuka, ketik **webauthn**.
3. Pilih `security.webauth.webauthn` dan ubah nilainya menjadi `true`.

- Apakah Anda menggunakan plugin peramban?

AWS tidak mendukung penggunaan plugin untuk menambahkan dukungan WebAuthn browser. Sebaliknya, gunakan browser yang menawarkan dukungan bawaan dari WebAuthn standar.

Meskipun Anda menggunakan peramban yang didukung, Anda mungkin memiliki plugin yang tidak kompatibel WebAuthn. Plugin yang tidak kompatibel mungkin mencegah Anda mengaktifkan dan menggunakan kunci keamanan yang sesuai dengan FIDO Anda. Anda harus menonaktifkan setiap plugin yang mungkin tidak kompatibel dan memulai ulang peramban Anda. Kemudian, coba aktifkan kembali tombol 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 Anda adalah administrator dan pengguna IAM Anda tidak dapat mengaktifkan kunci keamanan FIDO mereka meskipun telah menggunakan konfigurasi yang didukung, pastikan mereka memiliki izin yang sesuai. Untuk contoh terperinci, lihat [Tutorial IAM: Izinkan pengguna untuk mengelola kredensi dan pengaturan MFA mereka](#).

Saya tidak dapat masuk menggunakan kunci keamanan FIDO saya

Jika Anda pengguna IAM dan tidak dapat masuk AWS Management Console menggunakan kunci keamanan FIDO Anda, pertama-tama lihat [Konfigurasi yang didukung untuk menggunakan kunci](#)

[keamanan FIDO](#). Jika Anda menggunakan konfigurasi yang didukung tetapi tidak dapat masuk, hubungi administrator sistem untuk mendapatkan bantuan.

Saya menghilangkan 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 memerlukan satu perangkat MFA untuk masuk ke perangkat MFAAWS Management Console. Mengganti tombol keamanan FIDO mirip dengan mengganti token TOTP keras. Untuk informasi tentang apa yang harus dilakukan jika Anda menghilangkan atau merusak perangkat MFA tipe apa pun, lihat [Bagaimana jika perangkat MFA hilang atau berhenti bekerja?](#).

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 [AWSSupport](#).

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 akun AWS saya](#)
- [Saya tidak dapat mengedit atau menghapus peran di Akun AWS](#)
- [Saya tidak berwenang untuk melakukan: iam: PassRole](#)
- [Mengapa saya tidak dapat mengasumsikan peran dengan sesi 12 jam? \(API AWS CLI, AWS\)](#)
- [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 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 kredensial ini. Ini memastikan bahwa Anda selalu memiliki seperangkat kredensial 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. [Mengubah kebijakan kepercayaan peran \(konsole\)](#)
- Ketika Anda mengambil peran menggunakan AWS Management Console, pastikan Anda menggunakan nama peran yang tepat. Nama peran peka huruf besar/kecil.
- Ketika Anda mengambil peran menggunakan API AWS STS atau AWS CLI, pastikan Anda menggunakan nama peran yang tepat di ARN. Nama peran peka huruf besar/kecil.
- 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 “Menandai 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 menelepon `AssumeRole` adalah entitas tepercaya untuk peran yang Anda asumsikan. 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 meminta identitas meneruskan string kustom yang mengidentifikasi orang atau aplikasi yang melakukan tindakan 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 akun AWS saya

Beberapa layanan AWS mengharuskan Anda menggunakan tipe 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 [Menggunakan peran terkait 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 yang terkait dengan layanan di akun Anda dengan mengunjungi laman Peran IAM di konsol IAM. Peran yang terkait dengan layanan muncul dengan (Peran yang terkait dengan layanan) di kolom Entitas tepercaya di 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 Peran yang Terkait dengan Layanan. 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 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. 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 yang terkait dengan layanan, lihat [AWS layanan yang bekerja dengan IAM](#) dan cari layanan yang memiliki Ya di kolom Peran yang Terkait dengan Layanan.

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 dapat mengasumsikan peran dengan sesi 12 jam? (API AWS CLI, AWS)

Saat Anda menggunakan API AWS STS `AssumeRole*` atau operasi `assume-role*` CLI untuk mengasumsikan 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 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 [Lihat pengaturan durasi sesi maksimum untuk peran](#).

Jika Anda menggunakan [rantai peran](#) (menggunakan peran untuk mengasumsikan 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 pengenalan ramah seperti nama perusahaan Anda yang dapat digunakan sebagai pengganti ID Akun AWS. 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. Saat Anda [meminta kredensial keamanan sementara](#) secara terprogram menggunakan AWS STS, Anda dapat secara opsional meneruskan [kebijakan sesi](#) inline atau terkelola. 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. Saat Anda menyiapkan beberapa lingkungan layanan AWS, Anda harus menentukan peran yang akan diambil oleh layanan. Dalam beberapa kasus, layanan tersebut menciptakan peran layanan dan kebijakannya di IAM untuk Anda. Meskipun Anda dapat memodifikasi atau menghapus peran layanan dan kebijakannya dari dalam IAM, AWS tidak merekomendasikan hal ini. 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 selanjutnya 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 AWS Management Console 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 Entitas 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 terpercaya. Jika dokumentasi untuk layanan atau fitur yang Anda gunakan tidak menyertakan petunjuk untuk mencantumkan layanan sebagai prinsipal terpercaya, 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 Yes (Ya) di kolom Service-linked roles (Peran terkait layanan).
3. Pilih tautan Ya untuk melihat dokumentasi peran terkait layanan untuk layanan tersebut.
4. Temukan bagian Izin peran terkait layanan untuk layanan tersebut untuk melihat [prinsipal 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-tugas berikut:

1. Buat peran IAM menggunakan ID akun Anda. Jangan lampirkan kebijakan atau beri izin apa pun. Untuk detailnya, lihat [Membuat peran untuk mendelegasikan 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 prinsipal menjadi 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

Gunakan informasi di sini untuk membantu Anda memecahkan dan memperbaiki akses yang ditolak atau masalah lain yang mungkin Anda hadapi saat bekerja dengan Amazon EC2 dan IAM.

Topik

- [Saat mencoba meluncurkan instans, saya tidak melihat peran yang diharapkan di 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 instans 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?](#)

Saat mencoba meluncurkan instans, saya tidak melihat peran yang diharapkan di daftar Peran IAM konsol Amazon EC2

Periksa 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, lihat “Izin yang Diperlukan untuk Menggunakan Peran dengan Amazon EC2” di [Menggunakan peran IAM untuk memberikan izin pada aplikasi yang berjalan di instans Amazon EC2](#). Untuk informasi tentang menambahkan izin ke pengguna, lihat [Mengelola 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 dengan menggunakan CLI atau API IAM, verifikasi bahwa Anda membuat profil instans dan menambahkan peran tersebut ke profil instans tersebut. Selain itu, jika Anda menamai peran dan profil instans Anda secara berbeda, Anda tidak akan melihat nama peran yang benar dalam daftar peran IAM 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 [Menggunakan profil instans](#).

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, lihat “Bagaimana Cara Memulai?” di [Menggunakan peran IAM untuk memberikan izin pada aplikasi yang berjalan di instans Amazon EC2..](#) Untuk informasi tentang menambahkan izin ke pengguna, lihat [Mengelola kebijakan IAM.](#)

Amazon EC2: Ketika saya mencoba meluncurkan instans dengan peran, saya mendapatkan kesalahan **AccessDenied**

Periksa 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 [Membuat Peran IAM.](#)

Untuk informasi selengkapnya tentang izin yang diperlukan untuk bekerja dengan peran, lihat “Bagaimana Cara Memulai?” di [Menggunakan peran IAM untuk memberikan izin pada aplikasi yang berjalan di instans Amazon EC2..](#) Untuk informasi tentang menambahkan izin ke pengguna, lihat [Mengelola 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 pada aplikasi yang berjalan di instans Amazon EC2..](#)

Jika Anda masih tidak dapat mengakses kredensial keamanan sementara Anda di instans EC2, periksa hal berikut:

- Dapatkah Anda mengakses bagian lain dari Instance Metadata Service (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 `DescribeInstances` API EC2 atau menggunakan perintah `aws ec2 describe-instances` CLI.

```
[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 ID berbeda, profil instans yang terlampir pada instans 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.

```
{"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 contoh kebijakan, lihat [Contoh Kasus untuk Kebijakan Bucket Amazon S3](#) di Panduan Pengguna Amazon Simple Storage Service.

Saya masuk sebagai pengguna Akun AWS utama; mengapa saya tidak dapat mengakses bucket Amazon S3 dalam 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 menyebutkan Pengguna root akun AWS penanggung jawab, akses pengguna utama ke bucket tersebut ditolak. Namun, sebagai pengguna utama, Anda masih dapat mengakses bucket. Untuk melakukannya, ubah kebijakan bucket untuk memungkinkan akses pengguna utama dari konsol Amazon S3 atau konsol Amazon S3 AWS CLI. Gunakan prinsipal berikut, menggantikan `123456789012` dengan ID Akun AWS.

```
"Principal": { "AWS": "arn:aws:iam::123456789012:root" }
```

Memecahkan masalah federasi SAML 2.0 dengan AWS

Gunakan informasi di sini untuk membantu Anda mendiagnosis dan memperbaiki masalah yang mungkin Anda temukan saat bekerja dengan SAML 2.0 dan federasi dengan IAM.

Topik

- [Kesalahan: Permintaan Anda menyertakan respons SAMP yang tidak valid. Untuk logout, klik di sini.](#)
- [Kesalahan: RoleSessionName diperlukan dalam AuthnResponse \(layanan: AWSSecurityTokenService; kode status: 400; kode kesalahan: InvalidIdentityToken\)](#)
- [Kesalahan: Tidak berwenang untuk melakukan sts: AssumeRoleWith SAMP \(layanan: AWSSecurityTokenService; kode status: 403; kode kesalahan:\) AccessDenied](#)
- [Kesalahan: RoleSessionName di AuthnResponse harus cocok \[A-Za-Z_0-9+=, .@-\] {2,64} \(layanan:; kode status: 400; kode kesalahan: AWSSecurityTokenService\) InvalidIdentityToken](#)

- [Kesalahan: Identitas Sumber harus cocok dengan \[A-Za-Z_0-9+=, .@-\] {2,64} dan tidak dimulai dengan "aws:" \(layanan:; kode status: 400; kode kesalahan: AWSSecurityTokenService\) InvalidIdentityToken](#)
- [Kesalahan: Tanda tangan respons tidak valid \(layanan: AWSSecurityTokenService; kode status: 400; kode kesalahan:\) InvalidIdentityToken](#)
- [Kesalahan: Gagal mengambil peran: Penerbit tidak hadir di penyedia tertentu \(layanan: AWSOpenIdDiscoveryService; kode status: 400; kode kesalahan: AuthSamlInvalidSamlResponseException\)](#)
- [Kesalahan: Tidak dapat mengurai metadata.](#)
- [Kesalahan: Penyedia yang ditentukan tidak ada.](#)
- [Kesalahan: Diminta DurationSeconds melebihi MaxSessionDuration set untuk peran ini.](#)
- [Kesalahan: Respons tidak berisi audiens yang diperlukan.](#)
- [Cara melihat respons SAML di peramban Anda untuk pemecahan masalah](#)

Kesalahan: Permintaan Anda menyertakan respons SAMP yang tidak valid. Untuk logout, 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

Untuk informasi selengkapnya, lihat [Konfigurasi pernyataan SAMP untuk respons otentikasi](#). Untuk melihat respons SAML di peramban Anda, ikuti langkah-langkah yang tercantum di [Cara melihat respons SAML di peramban Anda untuk pemecahan masalah](#).

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 [Konfigurasi pernyataan SAMP untuk respons otentikasi](#). Untuk melihat respons SAML di peramban Anda, ikuti langkah-langkah yang tercantum di [Cara melihat respons SAML di peramban Anda untuk pemecahan masalah](#).

Kesalahan: Tidak berwenang untuk melakukan sts: AssumeRoleWith SAMP (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 dikonfigurasi untuk menggunakan [atribut PrincipalTag](#), kebijakan kepercayaan Anda juga harus mencakup tindakan `sts:TagSession`. Untuk informasi selengkapnya tentang tanda sesi, lihat [Melewati 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 dikonfigurasi 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 pengguna gabungan tidak memiliki izin untuk mengambil peran tersebut. Peran tersebut harus memiliki kebijakan kepercayaan yang menentukan ARN dari penyedia identitas SAMP IAM 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 [Menetapkan Izin di AWS untuk Pengguna Gabungan](#) dan [Konfigurasi pernyataan SAMP untuk respons otentikasi](#). Untuk melihat respons SAML di peramban Anda, ikuti langkah-langkah yang tercantum di [Cara melihat respons SAML di peramban Anda untuk pemecahan masalah](#).

Kesalahan: RoleSessionName di AuthnResponse harus cocok [A-Za-Z_0-9+=, .@-] {2,64} (layanan:; kode status: 400; kode kesalahan: AWSSecurityTokenService) 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 [Konfigurasi pernyataan SAMP untuk respons otentikasi](#). Untuk melihat respons SAML di peramban Anda, ikuti langkah-langkah yang tercantum di [Cara melihat respons SAML di peramban Anda untuk pemecahan masalah](#).

Kesalahan: Identitas Sumber harus cocok dengan [A-Za-Z_0-9+=, .@-] {2,64} dan tidak dimulai dengan "aws:" (layanan:; kode status: 400; kode kesalahan: AWSSecurityTokenService) InvalidIdentityToken

Kesalahan ini dapat terjadi jika nilai atribut sourceIdentity terlalu panjang atau mengandung karakter yang tidak valid. Panjang valid maksimum adalah 64 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 [Konfigurasi pernyataan SAMP untuk respons otentikasi](#). Untuk melihat respons SAML di peramban Anda, ikuti langkah-langkah yang tercantum di [Cara melihat respons SAML di peramban Anda untuk pemecahan masalah](#).

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 `aws iam update-saml-provider` CLI lintas platform atau cmdlet. `Update-IAMSAMLProvider` PowerShell

Kesalahan: Gagal mengambil peran: Penerbit tidak hadir 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 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 AWS Management Console, 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 SAMP (pernyataan) yang diterima dari IDP. File metadata harus diekode dalam format UTF-8 tanpa tanda urutan byte (BOM). Untuk menghapus BOM, Anda dapat mengkode file sebagai UTF-8 menggunakan alat bantu edit teks, seperti Notepad++.

Sertifikat x.509 yang disertakan sebagai bagian dari dokumen metadata SAMP harus menggunakan ukuran kunci minimal 1024 bit. Selain itu, sertifikat x.509 juga harus bebas dari ekstensi berulang. Anda dapat menggunakan ekstensi, tetapi ekstensi hanya dapat muncul sekali dalam sertifikat. Jika sertifikat x.509 tidak memenuhi salah satu kondisi, pembuatan IDP gagal dan mengembalikan kesalahan "Tidak dapat mengurai metadata".

Seperti yang didefinisikan oleh [SAMP V2.0 Metadata Interoperability Profile Versi 1.0](#), IAM tidak mengevaluasi atau mengambil tindakan terkait berakhirnya sertifikat X.509 dokumen metadata.

Kesalahan: Penyedia yang ditentukan tidak ada.

Kesalahan ini dapat terjadi jika nama penyedia yang Anda tentukan dalam pernyataan SAML tidak cocok dengan nama penyedia yang dikonfigurasi dalam IAM. Untuk informasi selengkapnya tentang melihat nama penyedia, lihat [Buat penyedia identitas SAMP di IAM](#).

Kesalahan: Diminta DurationSeconds melebihi MaxSessionDuration set untuk peran ini.

Kesalahan ini dapat terjadi jika Anda mengambil peran dari AWS CLI atau API.

Saat Anda menggunakan operasi [assume-role-with-saml](#) CLI atau [AssumeRoleWithSAMP](#) API untuk mengambil peran, Anda dapat menentukan nilai untuk parameter tersebut. `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 [Lihat pengaturan durasi sesi maksimum untuk peran](#).

Kesalahan: Respons tidak berisi audiens yang diperlukan.

Kesalahan ini dapat terjadi jika ada ketidakcocokan antara URL audiens dan penyedia identitas dalam konfigurasi SAMP. Pastikan bahwa pengenal pihak yang mengandalkan penyedia identitas (IDP) Anda sama persis dengan URL audiens (ID entitas) yang disediakan dalam konfigurasi SAMP.

Cara melihat respons SAML di peramban Anda untuk pemecahan masalah

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 SAML berkode Base64](#)

Google Chrome

Untuk melihat respons SAML di Chrome

Langkah-langkah ini diuji menggunakan versi 106.0.5249.103 (Build Resmi) (arm64) Google Chrome. Jika Anda menggunakan versi lain, Anda mungkin perlu mengadaptasi langkah-langkah tersebut.

1. Tekan F12 untuk memulai konsol Developer Tools.
2. Pilih tab Jaringan, dan kemudian pilih Preserve log di kiri atas jendela Developer Tools.
3. Mereproduksi masalah.
4. (Opsional) Jika kolom Metode tidak terlihat di panel log Developer Tools Network, klik kanan pada label kolom mana pun dan pilih Metode untuk menambahkan kolom.
5. Cari posting SALL di panel log Developer Tools Network. Pilih baris tersebut, lalu lihat tab Payload di bagian atas. Cari elemen SAMLResponse yang berisi permintaan terenkode. Nilai yang terkait adalah respons yang dienkodkan dengan Base64.

Mozilla Firefox

Untuk melihat respons SAML di Firefox

Prosedur ini diuji pada Mozilla Firefox versi 105.0.3 (64-bit). Jika Anda menggunakan versi lain, Anda mungkin perlu mengadaptasi langkah-langkah tersebut.

1. Tekan F12 untuk memulai konsol Alat Pengembang Web.
2. Pilih tab Jaringan.
3. Di kanan atas jendela alat developer web, pilih opsi (ikon roda gigi kecil). Pilih Bertahan log.
4. Mereproduksi masalah.
5. (Opsional) Jika kolom Metode tidak terlihat di panel log Jaringan Alat Pengembang Web, klik kanan pada label kolom apa pun dan pilih Metode untuk menambahkan kolom.
6. Cari POST SAML di dalam tabel. Pilih baris itu, lalu lihat tab Request dan temukan elemen SAMLResponse. Nilai yang terkait adalah respons yang dienkodkan dengan Base64.

Apple Safari

Untuk melihat respons SAML dalam Safari

Langkah-langkah ini diuji menggunakan Apple Safari versi 16.0 (17614.1.25.9.10, 17614). 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 Tampilkan Inspector Web.
3. Pilih tab Jaringan.
4. Di kiri atas jendela Web Inspector, pilih opsi (ikon lingkaran kecil yang berisi tiga garis horizontal). Pilih Preserve Log.
5. (Opsional) Jika kolom Metode tidak terlihat di panel log Jaringan Inspector Web, klik kanan pada label kolom mana pun dan pilih Metode untuk menambahkan kolom.
6. Mereproduksi masalah.
7. Cari POST SAML di dalam tabel. Pilih baris tersebut, lalu lihat tab Header.
8. Cari elemen SAMLResponse yang berisi permintaan terenkode. Gulir ke bawah untuk menemukan Request Data dengan nama SAMLResponse. Nilai yang terkait adalah respons yang dienkodkan dengan Base64.

Apa yang harus dilakukan dengan respons SAML berkode Base64

Setelah Anda menemukan elemen respons SAML berkode Base64 di peramban Anda, salin dan gunakan alat dekode Base-64 favorit Anda untuk mengekstrak respons yang ditandai dengan 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.

Opsi bawaan untuk sistem Windows (PowerShell):

```
PS C:
```

```
\> [System.Text.Encoding]::UTF8.GetString([System.Convert]::FromBase64String("base64encodedtext"))
```

Opsi bawaan untuk sistem macOS dan Linux:

```
$ echo "base64encodedtext" | base64 --decode
```


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

- [Amazon Resource Name \(ARN\)](#)
- [Pengidentifikasi IAM](#)
- [IAM dan AWS STS kuota](#)
- [Titik akhir VPC antarmuka](#)
- [AWS layanan yang bekerja dengan IAM](#)
- [Menandatangani permintaan AWS API](#)
- [Referensi kebijakan IAM JSON](#)

Amazon Resource Name (ARN)

Amazon Resource Names (ARN) mengidentifikasi sumber daya secara unik. AWS Kami memerlukan ARN saat Anda perlu menentukan sumber daya secara jelas di semua AWS, seperti dalam kebijakan IAM, tag Amazon Relational Database Service (Amazon RDS), dan panggilan API.

Format ARN

Berikut ini adalah format umum untuk ARN. Format spesifik bergantung pada sumber daya. Untuk menggunakan ARN, ganti teks yang *dicetak miring* 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 AWS Wilayah. Setiap AWS akun dicakup ke satu partisi.

Berikut ini adalah partisi yang didukung:

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

service

Namespace layanan yang mengidentifikasi produk. AWS

region

Kode Wilayah. Misalnya, `us-east-2` untuk US East (Ohio). Untuk daftar kode Region, lihat [Titik 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 qualifier seperti versi (resource-type:resource-name:qualifier).

Contoh-contoh

Pengguna IAM

```
arn:aws:iam:: 123456789012: pengguna/johndoe
```

Topik SNS

```
arn:aws:sns: us-timur-1:123456789012: example-sns-topic-name
```

VPC

```
arn:aws:ec2: us-timur-1:123456789012: vpc/vpc-0e9801d129Contoh
```

Cari format ARN untuk sumber daya

Format yang tepat dari ARN tergantung pada layanan dan jenis 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 arahkan ke tabel jenis sumber daya.

Jalur di ARN

ARN Sumber Daya dapat menyertakan jalur. Misalnya, di Amazon S3, pengidentifikasi 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 diizinkan di jalur IAM: garis miring maju (/), plus (+), sama dengan (=), koma (,), periode (.), at (@), garis bawah (_), dan @ tanda hubung (-). _ -

Menggunakan wildcard di jalur

Jalur dapat menyertakan karakter wildcard, yaitu tanda bintang (*). Misalnya, jika Anda menulis kebijakan IAM, Anda dapat menentukan semua pengguna IAM yang memiliki jalur `product_1234` menggunakan wildcard sebagai berikut:

```
arn:aws:iam::123456789012:user/Development/product_1234/*
```

Demikian pula, Anda dapat menentukan `user/*` yang berarti semua pengguna atau `group/*` yang berarti semua grup, seperti dalam contoh berikut:

```
"Resource": "arn:aws:iam::123456789012:user/*"  
"Resource": "arn:aws:iam::123456789012:group/*"
```

Contoh berikut menunjukkan ARN untuk bucket Amazon S3 di mana nama sumber daya menyertakan jalur:

```
arn:aws:s3::my_corporate_bucket/*  
arn:aws:s3::my_corporate_bucket/Development/*
```

Penggunaan wildcard salah

Anda tidak dapat menggunakan wildcard di bagian ARN yang menentukan tipe sumber daya, seperti istilah `user` dalam IAM ARN. Misalnya, hal berikut tidak diizinkan.

```
arn:aws:iam::123456789012:u* <== not allowed
```

Pengidentifikasi IAM

IAM menggunakan beberapa pengidentifikasi yang berbeda untuk pengguna, grup pengguna, 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](#)

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: Mengakses 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 AWS CloudFormation untuk membuat sumber daya, Anda dapat menentukan jalur untuk pengguna, grup pengguna, dan peran, serta grup 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 jalurnya 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 pengguna berdasarkan jalur mereka. Pengguna dengan jalur yang berbeda dapat menggunakan sumber daya yang sama jika mereka telah diberikan izin untuk 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 yang 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 partisinya 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 produk AWS. 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: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
```

Banyak dari contoh-contoh berikut mencakup jalur di bagian sumber daya ARN. Jalur tidak dapat dibuat atau dimanipulasi di AWS Management Console. Untuk menggunakan jalur, Anda harus bekerja dengan sumber daya dengan menggunakan AWS API, the AWS CLI, atau Tools untuk Windows PowerShell.


Contoh:

```
arn:aws:iam::123456789012:root
arn:aws:iam::123456789012:user/JohnDoe
arn:aws:iam::123456789012:user/division_abc/subdivision_xyz/JaneDoe
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/JohnDoe
arn:aws:sts::123456789012:assumed-role/Accounting-Role/JaneDoe
arn:aws:iam::123456789012:mfa/JaneDoeMFA
arn:aws:iam::123456789012:u2f/user/JohnDoe/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/ADFSPProvider
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
```

Contoh berikut memberikan detail lebih lanjut untuk membantu Anda memahami format ARN untuk berbagai jenis IAM dan sumber daya. AWS STS

- 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/JohnDoe
```

- Pengguna lain dengan jalur yang mencerminkan bagan organisasi:

```
arn:aws:iam::123456789012:user/division_abc/subdivision_xyz/JaneDoe
```

- 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
```

- Seorang pengguna federasi diidentifikasi dalam IAM sebagai "Paulo":

```
arn:aws:sts::123456789012:federated-user/Paulo
```

- Sesi aktif seseorang yang mengambil peran "Akuntansi-Peran", dengan nama sesi peran "Mary":

```
arn:aws:sts::123456789012:assumed-role/Accounting-Role/Mary
```

- 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/ADFSPProvider  
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 Amazon EKS
OIDC:

```
arn:aws:iam::123456789012:oidc-provider/oidc.eks.us-west-2.amazonaws.com/id/  
a1b2c3d4567890abcdefEXAMPLE11111
```


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 [Principalelemen](#) kebijakan berbasis sumber daya.

- Akun AWSMenampilkan yang berikut ini:

```
arn:aws:iam::123456789012:root
```

Contoh berikut menunjukkan kebijakan yang dapat Anda tetapkan kepada Richard untuk mengizinkannya mengelola kunci aksesnya. Perhatikan bahwa sumber daya adalah Richard, pengguna IAM.

```
{
  "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

Bila Anda 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 bagian *sumber daya* ARN untuk menentukan beberapa pengguna atau grup pengguna atau pun 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 `stringapp_`, Anda dapat merujuk mereka semua dengan ARN berikut.

```
arn:aws:iam::123456789012:user/division_abc/subdivision_xyz/product_1234/app_*
```

Untuk menentukan semua pengguna, grup pengguna, atau kebijakan dalam AndaAkun 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/JohnDoe
arn:aws:iam::111122223333:user/division_abc/subdivision_xyz/JaneDoe
```

Namun, jika Anda menentukan ARN berikut untuk pengguna `arn:aws:iam::111122223333:user/division_abc*`, ini cocok dengan contoh kedua, tetapi bukan yang pertama.

```
arn:aws:iam::111122223333:user/JohnDoe
arn:aws:iam::111122223333:user/division_abc/subdivision_xyz/JaneDoe
```

Jangan gunakan wildcard di `user/`, `group/`, atau `policy/` bagian dari ARN. Misalnya, IAM tidak mengizinkan hal berikut:

```
arn:aws:iam::123456789012:u*
```

Example Contoh penggunaan jalur dan ARN untuk grup pengguna berbasis proyek

Jalur tidak dapat dibuat atau dimanipulasi di AWS Management Console. Untuk menggunakan jalur, Anda harus bekerja dengan sumber daya dengan menggunakan AWS API, theAWS CLI, atau Tools untuk Windows PowerShell.

Dalam contoh ini, Jules dalam grup pengguna Marketing_Admin 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, sehingga Jules membuat grup pengguna baru dalam 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.

```
{
  "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

Ketika IAM membuat pengguna, grup pengguna, peran, kebijakan, profil instance, atau sertifikat server, IAM menetapkan ID unik untuk setiap sumber daya. ID unik terlihat seperti ini:

AIDAJQABLZS4A3QDU576Q

Untuk sebagian besar, Anda menggunakan nama dan [ARN](#) yang ramah saat Anda 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-nama ramah di AndaAkun AWS. Dalam akun Anda, nama ramah untuk pengguna, grup pengguna, peran, atau kebijakan harus unik. Misalnya, Anda dapat membuat nama John pengguna IAM. 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 cara membuat kebijakan berbasis identitas yang memungkinkan pengguna IAM mengakses objek bucket mereka sendiri di S3 menggunakan nama pengguna yang ramah, lihat [Amazon S3: Mengizinkan akses pengguna IAM untuk mengakses direktori rumah S3 mereka, secara terprogram dan di konsol](#)

Misalkan karyawan bernama John meninggalkan perusahaan Anda dan Anda menghapus nama John pengguna IAM yang sesuai. Tapi 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 untuk mengakses informasi yang ditinggalkan oleh John sebelumnya.

Secara umum, kami menyarankan 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 dengan ID unik dan tidak hanya dengan 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 [Principalelemen kebijakan](#) berbasis sumber daya.

```
"Principal": {
  "AWS": [
    "arn:aws:iam::111122223333:role/role-name",
```

```
"AIDACKCEVSQ6C2EXAMPLE",
"AROADBQP57FF2AEXAMPLE"
}
```

Contoh berikut menunjukkan cara menentukan ID unik dalam [Conditionelemen](#) 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 toko lain) dari pengguna IAM atau informasi peran. ID unik dapat memberikan pengenal unik untuk setiap pengguna IAM atau peran yang Anda buat. Ini adalah kasus ketika Anda memiliki pengguna IAM atau peran 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 STToken pembawa layanan
ACCA	Kredensial spesifik konteks
AGPA	Grup pengguna
AIDA	Pengguna IAM
AIPA	Profil instans Amazon EC2
AKIA	Access key

Awalan	Tipe sumber daya
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:

- [get-caller-identity](#)
- [dapatkan-kelompok](#)
- [dapatkan-peran](#)
- [dapatkan-pengguna](#)
- [dapatkan-kebijakan](#)
- [get-instance-profile](#)
- [get-server-certificate](#)

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](#)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 (,), periode (.), 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 [Cara](#)

[menggunakan ID eksternal saat memberikan akses ke sumber daya AWS Anda kepada pihak ketiga.](#)

- Nama kebijakan untuk [kebijakan sebaris](#) harus unik untuk pengguna, grup, atau peran yang disematkan. Nama-nama dapat berisi karakter Latin Dasar (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 bisa berupa angka 12 digit.

Untuk daftar karakter Latin Basic (ASCII), kunjungi [Tabel Kode Basic Kongres Latin \(ASCII\)](#).

Kuota objek IAM

Kuota, juga disebut sebagai batas dalam AWS, adalah nilai maksimum untuk sumber daya, tindakan, dan item dalam Akun AWS. Gunakan Service Quotas 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 Pengguna AWS Masuk untuk masuk ke AWS Management Console
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 Service Quotas.](#)

Anda dapat meminta peningkatan kuota default untuk kuota IAM yang dapat disesuaikan. Permintaan hingga permintaan [maximum quota](#) secara otomatis disetujui dan diselesaikan dalam beberapa menit.

Tabel berikut mencantumkan sumber daya yang area peningkatan kuota dapat disetujui secara otomatis.

Kuota yang dapat disesuaikan untuk sumber daya IAM

Sumber daya	Kuota default	Kuota maksimum
Kebijakan yang dikelola pelanggan dalam akun	1500	5000
Grup kunci per akun	300	500
Profil instans per akun	1000	5000
Kebijakan terkelola per peran	10	20
Kebijakan terkelola per pengguna	10	20
Panjang kebijakan kepercayaan peran	2048 karakter	4096 karakter
Tabel Per Akun	1000	5000
Sertifikat server per akun	20	1000

Kuota Penganalisis Akses IAM


Untuk daftar titik akhir layanan dan kuota layanan IAM Access Analyzer, lihat titik akhir dan kuota [IAM Access Analyzer](#) di. Referensi Umum AWS



Kuota Peran IAM Di Mana Saja

Untuk daftar titik akhir layanan dan kuota layanan IAM Roles Anywhere, lihat Titik akhir dan kuota [AWS Identity and Access Management Peran Di Mana Saja](#) di. Referensi Umum AWS

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 batasan berikut.


Deskripsi	Kuota
Alias untuk ID Akun AWS	3–63 karakter
Untuk kebijakan inline	<p>Anda dapat menambahkan kebijakan inline sebanyak yang Anda inginkan kepada pengguna, peran, atau grup IAM. Tetapi ukuran total kebijakan agregat (ukuran jumlah semua kebijakan sebaris) per entitas tidak dapat melebihi batas berikut:</p> <ul style="list-style-type: none">• Ukuran kebijakan pengguna tidak boleh melebihi 2.048 karakter.• Ukuran kebijakan peran tidak dapat melebihi 10.240 karakter.• Ukuran kebijakan grup tidak boleh melebihi 5.120 karakter. <div data-bbox="829 1247 1507 1514"><p> Note</p><p>IAM tidak menghitung ruang putih saat menghitung ukuran kebijakan terhadap batas-batas ini.</p></div>
Untuk kebijakan terkelola	<ul style="list-style-type: none">• Ukuran setiap kebijakan terkelola tidak boleh melebihi 6.144 karakter.

Deskripsi	Kuota
	<p> Note</p> <p>IAM tidak menghitung spasi putih saat menghitung ukuran kebijakan terhadap batas ini.</p>
Nama grup	128 karakter
Nama profil instans	128 karakter
Kata sandi untuk profil masuk	1–128 karakter
Jalur	512 karakter
Nama kebijakan	128 karakter
Nama peran	64 karakter
	<p> Important</p> <p>Jika Anda ingin menggunakan peran dengan fitur Switch Role diAWS Management Console, maka gabungan Path dan tidak roleName dapat melebihi 64 karakter.</p>

Deskripsi	Kuota
Durasi sesi peran	<p>12 jam</p> <p>Saat Anda mengasumsikan peran dari AWS CLI atau API, Anda dapat menggunakan parameter <code>duration-seconds</code> CLI atau parameter API <code>DurationSeconds</code> untuk meminta sesi 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 menentukan nilai untuk parameter <code>DurationSeconds</code> 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 yang lebih sedikit. Pengaturan durasi sesi maksimum tidak membatasi sesi yang diasumsikan oleh AWS layanan. Untuk mempelajari cara melihat nilai maksimum untuk peran Anda, lihat Lihat pengaturan durasi sesi maksimum untuk peran.</p>
Nama sesi peran	64 karakter

Deskripsi	Kuota
Peran kebijakan sesi	<ul style="list-style-type: none">• Ukuran dokumen kebijakan JSON yang diteruskan dan semua karakter ARN kebijakan terkelola yang diteruskan digabungkan tidak dapat 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 secara terprogram membuat sesi sementara untuk peran atau pengguna federasi.• Selain itu, AWS konversi memampatkan kebijakan sesi dan tag sesi yang diteruskan ke dalam format biner yang dikemas yang memiliki batas terpisah. Elemen respons <code>PackedPolicySize</code> menunjukkan persentase seberapa dekat kebijakan dan tanda untuk permintaan Anda dengan batas ukuran atas.• Sebaiknya teruskan kebijakan sesi menggunakan AWS CLI atau API AWS. AWS Management Console mungkin menambahkan informasi sesi konsol tambahan ke kebijakan yang dikemas.

Deskripsi	Kuota
Peran tanda sesi	<ul style="list-style-type: none">• Tag sesi harus memenuhi batas kunci tag 128 karakter dan batas nilai tag 256 karakter.• Anda dapat meneruskan hingga 50 tanda sesi.• Konversi AWS mengompresi kebijakan sesi yang lolos dan tanda sesi ke dalam format biner yang dikemas yang memiliki batas terpisah. Anda dapat meneruskan tanda sesi menggunakan AWS CLI atau API AWS. Elemen respons <code>PackedPolicySize</code> menunjukkan persentase seberapa dekat kebijakan dan tanda untuk permintaan Anda dengan batas ukuran atas.
Saml otentikasi respon base64 dikodekan	100.000 karakter Batas karakter ini berlaku untuk assume-role-with-saml operasi 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 bisa kosong yang berarti nilai tag dapat memiliki panjang 0 karakter.

Deskripsi	Kuota
ID unik yang dibuat oleh IAM	<p>128 karakter. Sebagai contoh:</p> <ul style="list-style-type: none">• 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 <div data-bbox="829 739 1507 1102"><p> Note</p><p>Ini tidak dimaksudkan untuk menjadi daftar lengkap, juga bukan jaminan bahwa ID dari jenis tertentu dimulai hanya dengan kombinasi huruf yang ditentukan.</p></div>
Nama pengguna	64 karakter

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 Security Token Service AWS STS Anda dapat menggunakan koneksi ini untuk memungkinkan 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 ke AWS STS, Anda menentukan antarmuka VPC endpoint untuk AWS STS. Endpoint menyediakan konektivitas yang andal dan dapat diskalakan 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.

Endpoint VPC antarmuka didukung oleh AWS PrivateLink, sebuah AWS teknologi yang memungkinkan komunikasi pribadi antara AWS layanan menggunakan antarmuka jaringan elastis dengan alamat IP pribadi. Untuk informasi selengkapnya, lihat [AWS PrivateLink AWS Layanan](#).

Informasi berikut adalah untuk pengguna Amazon VPC. Untuk informasi selengkapnya, lihat [Memulai dengan Amazon VPC](#) dalam Panduan Pengguna Amazon VPC.

Ketersediaan

AWS STS saat ini mendukung titik akhir VPC di Wilayah berikut:

- AS Timur (Ohio)
- AS Timur (Virginia Utara)
- AS Barat (California Utara)
- AS Barat (Oregon)
- Afrika (Cape Town)
- Asia Pasifik (Hong Kong)
- Asia Pasifik (Mumbai)
- Asia Pasifik (Osaka)
- Asia Pasifik (Seoul)
- Asia Pasifik (Singapura)
- Asia Pasifik (Sydney)
- Asia Pasifik (Tokyo)
- Kanada (Pusat)
- Tiongkok (Beijing)
- Tiongkok (Ningxia)
- Eropa (Frankfurt)
- Eropa (Irlandia)
- Eropa (London)
- Eropa (Milan)

- Eropa (Paris)
- Eropa (Stockholm)
- Timur Tengah (Bahrain)
- Amerika Selatan (Sao Paulo)
- AWS GovCloud (AS-Timur)
- AWS GovCloud (AS-Barat)

Buat titik akhir VPC untuk AWS STS

Untuk mulai menggunakan AWS STS dengan VPC Anda, buat antarmuka VPC endpoint untuk AWS STS. Untuk informasi selengkapnya, lihat [Mengakses AWS layanan menggunakan titik akhir VPC antarmuka di Panduan Pengguna Amazon VPC](#).

Setelah Anda membuat titik akhir VPC, Anda harus menggunakan titik akhir regional yang cocok untuk mengirim permintaan Anda. AWS STS merekomendasikan agar Anda menggunakan kedua metode `setEndpoint` dan `setRegion` 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 endpoint 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 kredensial sementara AWS STS dari sumber daya yang berada 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, lihat [Mengelola AWS STS dalam Wilayah AWS](#).

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](#).
- Izin tingkat sumber daya — 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.
- Kebijakan berbasis sumber daya – Anda dapat melampirkan kebijakan berbasis sumber daya pada sumber daya dalam layanan. Kebijakan berbasis sumber daya mencakup elemen Principal untuk menentukan identitas IAM mana yang dapat mengakses sumber daya tersebut. Untuk informasi selengkapnya, lihat [Kebijakan berbasis identitas 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 hanya untuk beberapa jenis sumber daya, nilainya adalah Parsial. Untuk informasi selengkapnya tentang mendefinisikan izin berdasarkan atribut seperti tanda, lihat [Untuk apa ABAC? AWS](#). Untuk melihat tutorial dengan langkah-langkah untuk menyiapkan ABAC, lihat [Menggunakan kontrol akses berbasis atribut \(ABAC\)](#).
- Kredensi sementara — Anda dapat menggunakan kredensial jangka pendek yang diperoleh saat masuk menggunakan Pusat Identitas IAM, 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 kredensial 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](#).
- Peran yang ditautkan dengan layanan – Sebuah [peran yang berkaitan dengan layanan](#) adalah jenis peran layanan khusus yang memberikan izin layanan untuk mengakses sumber daya pada 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 [Menggunakan peran terkait layanan](#).

- 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	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensia I sementara	Peran terkait layanan
AWS Account Management	Ya	Ya	Tidak	Tidak	Ya	Tidak
AWS Activate Console	Ya	Tidak	Tidak	Tidak	Ya	Tidak
AWS Amplify Admin	Ya	Ya	Tidak	Tidak	Ya	Tidak
AWS Amplify	Ya	Ya	Tidak	Sebagai	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

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
Amazon API Gateway	Ya	Ya	Ya	Tida	Ya	<u>Ya</u>
Manajemen Amazon API Gateway	Ya	Ya	Tida	Ya	Ya	Tidak
Manajemen Amazon API Gateway V2	Ya	Ya	Tida	Ya	Ya	Tidak
AWS App2Container	Ya	Tida	Tida	Tida	Ya	Tidak
AWS AppConfig	Ya	Ya	Tida	Ya	Ya	Tidak
AWS AppFabric	Ya	Ya	Tida	Ya	Ya	Tidak
Amazon AppFlow	Ya	Ya	Tida	Ya	Ya	Tidak
Amazon AppIntegrations	Ya	Ya	Tida	Ya	Ya	<u>Ya</u>

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensia I sementara	Peran terkait layanan
Application Auto Scaling	Ya	Ya	Tida	Ya	Ya	Ya
AWS Profiler Biaya Aplikasi	Ya	Tida	Tida	Tida	Ya	Tidak
AWS Aplikasi Discovery Arsenal	Ya	Tida	Tida	Tida	Ya	Tidak
AWS Application Discovery Service	Ya	Tida	Tida	Tida	Ya	Ya
AWS Application Migration Service	Ya	Ya	Tida	Ya	Ya	Ya
AWS Layanan Transformasi Aplikasi	Ya	Tida	Tida	Tida	Ya	Tidak
AWS App Mesh	Ya	Ya	Tida	Ya	Ya	Ya
AWS App Mesh Pratinjau	Ya	Ya	Tida	Tida	Ya	Ya

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensia I sementara	Peran terkait layanan
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
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
AWS Auto Scaling	Ya	Tidak	Tidak	Tidak	Ya	Ya
AWS Pertukaran Data B2B	Ya	Ya	Tidak	Ya	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensia l sementara	Peran terkait layanan
AWS Backup	Ya	Ya	Ya	Ya	Ya	Ya
AWS Backup Gerbang	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Backup penyimpanan	Ya	Tida	Tida	Tida	Ya	Tidak
AWS Batch	Ya	Sebag	Tida	Ya	Ya	Ya
Batuan Dasar Amazon	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Billing and Cost Management	Ya	Tida	Tida	Tida	Ya	Ya
AWS Billing and Cost Management Ekspor Data	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Billing Conductor	Ya	Ya	Tida	Ya	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensia I sementara	Peran terkait layanan
Amazon Braket	Ya	Ya	Tida	Ya	Ya	Ya
AWS Layanan Anggaran	Ya	Ya	Tida	Tida	Tida	Tidak
AWS BugBust	Ya	Ya	Tida	Ya	Ya	Ya
AWS Certificate Manager (ACM)	Ya	Ya	Tida	Ya	Ya	Ya
AWS Chatbot	Ya	Ya	Tida	Tida	Ya	Ya
Amazon Chime	Ya	Ya	Tida	Ya	Ya	Ya
AWS Clean Rooms	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Clean Rooms ML	Ya	Ya	Tida	Ya	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensia I sementara	Peran terkait layanan
AWS Client VPN	Ya	Ya	Tida	Tida	Ya	Ya
AWS Cloud9	Ya	Ya	Ya	Ya	Ya	Ya
AWS Cloud Kontrol API	Ya	Tida	Tida	Tida	Ya	Tidak
Amazon Cloud Directory	Ya	Ya	Tida	Tida	Ya	Tidak
AWS CloudFormation	Ya	Ya	Tida	Ya	Ya	Tidak
Amazon CloudFront	Ya	Ya	Tida	Ya	Ya	Sebagian (Info)
Amazon CloudFront Key Value Store	Ya	Ya	Tida	Tida	Ya	Tidak
AWS CloudHSM	Ya	Ya	Tida	Ya	Ya	Ya

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
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)	Sebagian (Info)	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

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
Amazon CloudWatch Terbukti	Ya	Ya	Tidak	Ya	Ya	Tidak
Monitor CloudWatch Internet Amazon	Ya	Ya	Tidak	Ya	Ya	Tidak
CloudWatch Log Amazon	Ya	Ya	Ya	Sebagai	Ya	Ya
Monitor CloudWatch Jaringan Amazon	Ya	Ya	Tidak	Ya	Ya	Tidak
Manajer Akses CloudWatch Observabilitas Amazon	Ya	Ya	Tidak	Ya	Ya	Tidak
Amazon CloudWatch RUM	Ya	Ya	Tidak	Ya	Ya	Tidak
Amazon CloudWatch Synthetics	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS CodeArtifact	Ya	Ya	Ya	Ya	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
AWS CodeBuild	Ya	Ya	Ya (Info)	Sebagai (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

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensia I sementara	Peran terkait layanan
CodeGuru Keamanan Amazon	Ya	Ya	Tida	Ya	Ya	Tidak
AWS CodePipeline	Ya	Sebag	Tida	Ya	Ya	Tidak
AWS CodeStar	Ya	Sebag	Tida	Ya	Ya	Tidak
AWS CodeStar Koneksi	Ya	Ya	Tida	Ya	Ya	Ya
AWS CodeStar Pemberitahuan	Ya	Ya	Tida	Ya	Ya	Ya
Amazon CodeWhisperer	Ya	Ya	Tida	Ya	Ya	Ya
Amazon Cognito	Ya	Ya	Tida	Ya	Ya	Ya
Sinkronisasi Amazon Cognito	Ya	Ya	Tida	Tida	Ya	Ya

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensia I sementara	Peran terkait layanan
Kumpulan pengguna Amazon Cognito	Ya	Ya	Tida	Ya	Ya	Ya
Amazon Comprehend	Ya	Ya	Tida	Ya	Ya	Tidak
Amazon Comprehend Medical	Ya	Tida	Tida	Tida	Ya	Tidak
AWS Compute Optimizer	Ya	Tida	Tida	Tida	Ya	Ya
AWS Config	Ya	Sebar (Info)	Tida	Ya	Ya	Ya
Amazon Connect	Ya	Ya	Tida	Ya	Ya	Ya
Kasus Amazon Connect	Ya	Ya	Tida	Ya	Ya	Tidak
Profil Pelanggan Amazon Connect	Ya	Ya	Tida	Ya	Ya	Ya

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensia I sementara	Peran terkait layanan
Amazon Connect Komunikasi keluar volume tinggi	Ya	Ya	Tidak	Ya	Ya	Tidak
ID Suara Amazon Connect	Ya	Ya	Tidak	Ya	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
AWS Cost and Usage Report	Ya	Ya	Tidak	Tidak	Ya	Tidak
AWS Cost Explorer	Ya	Ya	Tidak	Ya	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
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	Tidak
Amazon Data Lifecycle Manager	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS Data Pipeline	Ya	Ya	Tidak	Sebag	Ya	Tidak
AWS DataSync	Ya	Ya	Tidak	Ya	Ya	Ya

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensia I sementara	Peran terkait layanan
Amazon DataZone	Ya	Tida	Tida	Tida	Ya	Tidak
AWS Batas Waktu Cloud	Ya	Ya	Tida	Ya	Ya	Tidak
AWS DeepComposer	Ya	Ya	Tida	Ya	Ya	Tidak
AWS DeepRacer	Ya	Ya	Tida	Ya	Ya	Ya
Detektif Amazon	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Device Farm	Ya	Ya	Tida	Ya	Ya	Ya
DevOpsGuru Amazon	Ya	Ya	Tida	Tida	Ya	Ya
AWS Alat diagnostik	Ya	Ya	Tida	Ya	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensia I sementara	Peran terkait layanan
AWS Direct Connect	Ya	Ya	Tida	Ya	Ya	Ya
AWS Directory Service	Ya	Ya	Tida	Ya	Ya	Tidak
Cluster Elastis Amazon DocumentDB	Ya	Ya	Tida	Ya	Ya	Tidak
Akselerator Amazon DynamoDB (DAX)	Ya	Ya	Tida	Tida	Ya	Ya
Amazon DynamoDB	Ya	Ya	Ya	Tida	Ya	Tidak
Amazon Elastic Compute Cloud (Amazon EC2)	Ya	Sebagian	Tida	Ya	Ya	Sebagian (Info)
Amazon EC2 Auto Scaling	Ya	Ya	Tida	Ya	Ya	Ya
EC2 Image Builder	Ya	Ya	Tida	Ya	Ya	Ya

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
Connect Instans Amazon EC2	Ya	Ya	Tida	Tida	Ya	Ya
Amazon ElastiCache	Ya	Ya	Tida	Ya	Ya	Ya
AWS Elastic Beanstalk	Ya	Sebagai	Tida	Ya	Ya	Ya
Toko Blok Elastis Amazon (Amazon EBS)	Ya	Sebagai	Tida	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	Tida	Ya	Ya	Tidak
Amazon Elastic Container Service (Amazon ECS)	Ya	Sebagai (Info)	Tida	Ya	Ya	Ya
AWS Elastic Disaster Recovery	Ya	Ya	Tida	Ya	Ya	Ya

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
Amazon Elastic File System (Amazon EFS)	Ya	Ya	Ya	Tidak	Ya	Ya
Amazon Elastic Inference	Ya	Ya	Tidak	Tidak	Ya	Tidak
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
Amazon Elastic Transcoder	Ya	Ya	Tidak	Tidak	Ya	Tidak
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

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensia l sementara	Peran terkait layanan
AWS Elemental MediaConnect	Ya	Ya	Tida	Tida	Ya	Ya
AWS Elemental MediaConvert	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Elemental MediaLive	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Elemental MediaPackage	Ya	Ya	Tida	Ya	Ya	Sebagian (Info)
AWS Elemental MediaPackage V2	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Elemental MediaPackage VOD	Ya	Ya	Tida	Ya	Ya	Sebagian (Info)
AWS Elemental MediaStore	Ya	Ya	Ya	Ya	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
AWS Elemental MediaTailor	Ya	Ya	Tida	Ya	Ya	Ya
AWS Kasus Dukungan Elemental	Ya	Tida	Tida	Tida	Ya	Tidak
AWS Konten Dukungan Elemental	Ya	Tida	Tida	Tida	Ya	Tidak
Amazon EMR	Ya	Ya	Tida	Ya	Ya	Ya
Amazon EMR di EKS	Ya	Ya	Tida	Ya	Ya	Ya
Amazon EMR Tanpa Server	Ya	Ya	Tida	Ya	Ya	Ya
AWS Resolusi Entitas	Ya	Ya	Tida	Ya	Ya	Tidak
Amazon EventBridge	Ya	Ya	Ya	Ya	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensia I sementara	Peran terkait layanan
EventBridge Pipa Amazon	Ya	Ya	Tida	Ya	Ya	Tidak
EventBridge Penjadwal Amazon	Ya	Ya	Tida	Ya	Ya	Tidak
EventBridge Skema Amazon	Ya	Ya	Ya	Ya	Ya	Tidak
AWS Layanan Injeksi Kesalahan	Ya	Ya	Tida	Ya	Ya	Ya
Amazon FinSpace	Ya	Ya	Tida	Ya	Ya	Ya
Amazon FinSpace API	Ya	Ya	Tida	Tida	Ya	Tidak
AWS Firewall Manager	Ya	Ya	Tida	Ya	Ya	Sebagian
Fleet Hub for AWS IoT Device Management	Ya	Ya	Tida	Ya	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensia l sementara	Peran terkait layanan
Amazon Forecast	Ya	Ya	Tida	Ya	Ya	Tidak
Amazon Fraud Detector	Ya	Ya	Tida	Ya	Ya	Tidak
FreeRTOS	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Tingkat Gratis	Ya	Tida	Tida	Tida	Ya	Tidak
Amazon FSx	Ya	Ya	Tida	Ya	Ya	Ya
Amazon GameLift	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Global Accelerator	Ya	Ya	Tida	Ya	Ya	Ya
AWS Glue	Ya	Ya	Ya	Sebag	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensia I sementara	Peran terkait layanan
AWS Glue DataBrew	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Ground Station	Ya	Ya	Tida	Ya	Ya	Ya
Pelabelan Ground Truth Amazon	Ya	Tida	Tida	Tida	Ya	Tidak
Amazon GuardDuty	Ya	Ya	Tida	Ya	Ya	Ya
AWS Health API Dan Pemberitahuan	Ya	Ya	Tida	Tida	Ya	Tidak
AWS HealthImaging	Ya	Ya	Tida	Ya	Ya	Tidak
AWS HealthLake	Ya	Ya	Tida	Ya	Ya	Tidak
AWS HealthOmics	Ya	Ya	Tida	Ya	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
Honeycode Amazon	Ya	Ya	Tidak	Tidak	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 Penganalisis Akses	Ya	Ya	Tidak	Ya	Ya	Sebagian
AWS Identity and Access Management Peran Di Mana Saja	Ya	Ya	Tidak	Ya	Ya	Ya

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensia l sementara	Peran terkait layanan
AWS Auth Toko Identitas	Ya	Tidak	Tidak	Tidak	Ya	Tidak
AWS Sinkronisasi Identitas	Ya	Ya	Tidak	Tidak	Ya	Tidak
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

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensia l sementara	Peran terkait layanan
AWS Invoicing	Ya	Tida	Tida	Tida	Ya	Tida
AWS IoT 1-Click	Ya	Ya	Tida	Ya	Ya	Tida
AWS IoT Analytics	Ya	Ya	Tida	Ya	Ya	Tida
AWS IoT	Ya	Ya	Sebag (Info)	Ya	Ya	Tida
AWS IoT Core Penasihat Perangkat	Ya	Ya	Tida	Ya	Ya	Tida
AWS IoT Penguji Perangkat	Ya	Tida	Tida	Tida	Ya	Tida
AWS IoT Events	Ya	Ya	Tida	Ya	Ya	Tida
AWS IoT FleetWise	Ya	Ya	Tida	Ya	Ya	Tida

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensia l sementara	Peran terkait layanan
AWS IoT Greengrass	Ya	Ya	Tida	Ya	Ya	Tidak
AWS IoT Greengrass V2	Ya	Ya	Tida	Sebag	Ya	Tidak
AWS IoT Lowongan DataPlane	Ya	Ya	Tida	Tida	Ya	Tidak
AWS IoT RoboRunner	Ya	Ya	Tida	Tida	Ya	Tidak
AWS IoT SiteWise	Ya	Ya	Tida	Ya	Ya	Ya
AWS IoT TwinMaker	Ya	Ya	Tida	Ya	Ya	Ya
AWS IoT Wireless	Ya	Ya	Tida	Ya	Ya	Tidak
AWS IQ	Ya	Ya	Tida	Tida	Ya	Ya

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
AWS Izin IQ	Ya	Ya	Tidak	Tidak	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

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensia l sementara	Peran terkait layanan
Amazon Kinesis Data Streams	Ya	Ya	Ya	Tida	Ya	Tida
Aliran Video Amazon Kinesis	Ya	Ya	Tida	Ya	Ya	Tida
AWS Lake Formation	Ya	Tida	Tida	Tida	Ya	Ya
AWS Lambda	Ya	Ya	Ya	Sebagian (Info)	Ya	Sebagian (Info)
AWS Launch Wizard	Ya	Tida	Tida	Tida	Ya	Tida
Amazon Lex	Ya	Ya	Tida	Ya	Ya	Ya
Amazon Lex V2	Ya	Ya	Ya	Ya	Ya	Ya
AWS License Manager	Ya	Ya	Tida	Ya	Ya	Ya

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
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
Amazon Lookout for Equipment	Ya	Ya	Tidak	Ya	Ya	Tidak
Amazon Lookout for Metrics	Ya	Ya	Tidak	Ya	Ya	Tidak
Amazon Lookout for Vision	Ya	Ya	Tidak	Ya	Ya	Tidak
Amazon Machine Learning	Ya	Ya	Tidak	Tidak	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensia I sementara	Peran terkait layanan
Amazon Macie	Ya	Ya	Tida	Ya	Ya	Ya
AWS Mainframe Modernization	Ya	Ya	Tida	Ya	Ya	Ya
Blockchain yang Dikelola Amazon	Ya	Ya	Tida	Ya	Ya	Tidak
Kueri Blockchain yang Dikelola Amazon	Ya	Tida	Tida	Tida	Ya	Tidak
Grafana yang Dikelola Amazon	Ya	Ya	Tida	Ya	Ya	Ya
Layanan Dikelola Amazon untuk Prometheus	Ya	Ya	Tida	Ya	Ya	Tidak
Amazon Managed Streaming for Apache Kafka (MSK)	Ya	Ya	Sebagai (Info)	Ya	Ya	Ya
Amazon Managed Streaming for Kafka Connect	Ya	Ya	Tida	Tida	Ya	Ya













Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
Alur Kerja Terkelola Amazon untuk Apache Airflow	Ya	Ya	Tidak	Ya	Ya	Tidak
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
AWS Marketplace Entitlement Service	Ya	Tidak	Tidak	Tidak	Ya	Tidak
AWS Marketplace Layanan Image Building	Ya	Tidak	Tidak	Tidak	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
Portal Manajemen AWS Marketplace	Ya	Tida	Tida	Tida	Ya	Tidak
AWS Marketplace Metering Service	Ya	Tida	Tida	Tida	Ya	Tidak
AWS Marketplace Marketplace Pribadi	Ya	Tida	Tida	Tida	Ya	Tidak
AWS Marketplace Integrasi Sistem Pengadaan	Ya	Tida	Tida	Tida	Ya	Tidak
AWS Marketplace Pelaporan Penjual	Ya	Ya	Tida	Tida	Ya	Tidak
AWS Marketplace Wawasan Vendor	Ya	Ya	Tida	Ya	Ya	Tidak
Amazon Mekanikal Turk	Ya	Tida	Tida	Tida	Ya	Tidak
Amazon MediaImport	Ya	Tida	Tida	Tida	Tida	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
Amazon MemoryDB for Redis	Ya	Ya	Tida	Ya	Ya	Ya
Layanan Pengiriman Pesan Amazon	Ya	Tida	Tida	Tida	Ya	Tidak
Layanan Gateway Pesan Amazon	Ya	Tida	Tida	Tida	Ya	Tidak
AWS Microservice Extractor for .NET	Ya	Tida	Tida	Tida	Ya	Tidak
AWS Kredit Program Percepatan Migrasi	Ya	Ya	Tida	Tida	Ya	Tidak
AWS Migration Hub	Ya	Ya	Tida	Tida	Ya	Ya
AWS Migration Hub Orkestrator	Ya	Ya	Tida	Ya	Ya	Ya
AWS Migration Hub Refactor Spaces	Ya	Ya	Ya	Ya	Ya	Ya

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensia l sementara	Peran terkait layanan
AWS Migration Hub Rekomendasi Strategi	Ya	Tida	Tida	Tida	Ya	Ya
Amazon Monitron	Ya	Ya	Tida	Ya	Ya	Ya
Amazon MQ	Ya	Ya	Tida	Ya	Ya	Ya
Amazon Neptunus	Ya	Ya	Tida	Tida	Ya	Ya
Analisis Amazon Neptunus	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Network Firewall	Ya	Ya	Tida	Ya	Ya	Ya
AWS Network Manager	Ya	Ya	Tida	Ya	Ya	Ya (Info)
AWS Network Manager Obrolan	Ya	Tida	Tida	Tida	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensia l sementara	Peran terkait layanan
Studio Amazon Nimble	Ya	Ya	Tida	Ya	Ya	Tidak
Amazon Satu Perusahaan	Ya	Ya	Tida	Ya	Ya	Tidak
OpenSearchTertelan Amazon	Ya	Ya	Tida	Ya	Ya	<u>Ya</u>
Amazon Tanpa OpenSearch Server	Ya	Ya	Tida	Ya	Ya	<u>Ya</u>
OpenSearch Layanan Amazon	Ya	Ya	Ya	Ya	Ya	<u>Ya</u>
AWS OpsWorks	Ya	Ya	Tida	Tida	Ya	Tidak
AWS OpsWorks Manajemen Konfigurasi	Ya	Ya	Tida	Tida	Ya	Tidak
AWS Organizations	Ya	Ya	Ya	Ya	Tida	<u>Ya</u>

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensia I sementara	Peran terkait layanan
AWS Outposts	 Ya	 Ya	 Tida	 Ya	 Ya	 Ya
AWS Panorama	 Ya	 Ya	 Tida	 Ya	 Ya	 Ya
AWS Partner Manajemen akun pusat	 Ya	 Tida	 Tida	 Tida	 Ya	 Tidak
AWS Payment Cryptography	 Ya	 Ya	 Tida	 Ya	 Ya	 Tidak
AWS Pembayaran	 Ya	 Tida	 Tida	 Tida	 Ya	 Tidak
AWS Wawasan Performa	 Ya	 Ya	 Tida	 Tida	 Ya	 Tidak
Amazon Personalisasi	 Ya	 Ya	 Tida	 Tida	 Ya	 Tidak
Amazon Pinpoint	 Ya	 Ya	 Tida	 Ya	 Ya	 Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensia I sementara	Peran terkait layanan
Layanan Email Amazon Pinpoint	Ya	Ya	Tida	Ya	Ya	Tidak
Amazon Pinpoint SMS dan Layanan Suara	Ya	Tida	Tida	Tida	Ya	Tidak
Amazon Pinpoint SMS dan Layanan Suara V2	Ya	Ya	Tida	Ya	Ya	Tidak
Amazon Polly	Ya	Ya	Tida	Tida	Ya	Tidak
Daftar Harga AWS	Ya	Tida	Tida	Tida	Ya	Tidak
AWS 5G pribadi	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Private CA Konekt or untuk Active Directory	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Private Certificate Authority (AWS Private CA)	Ya	Ya	Ya	Ya	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensia I sementara	Peran terkait layanan
AWS Proton	Ya	Ya	Tida	Ya	Ya	Ya
AWS Konsol Pesanan Pembelian	Ya	Ya	Tida	Ya	Ya	Tidak
Amazon Q	Ya	Tida	Tida	Tida	Ya	Tidak
Amazon Q Bisnis	Ya	Ya	Tida	Ya	Ya	Tidak
Amazon Q di Connect	Ya	Ya	Tida	Ya	Ya	Tidak
Database Buku Besar Amazon Quantum (Amazon QLDB)	Ya	Ya	Tida	Ya	Ya	Tidak
Amazon QuickSight	Ya	Ya	Tida	Ya	Ya	Tidak
API Data Amazon RDS	Ya	Ya	Tida	Tida	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
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	Ya	Ya	Tidak
Amazon Rekognition	Ya	Ya	Sebagai (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

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
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
AWS Resource Groups	Ya	Ya	Tidak	Ya	Sebagai (Info)	Tidak
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

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensia I sementara	Peran terkait layanan
Pengontrol Pemulihan Aplikasi Amazon Route 53 - Zonal Shift	Ya	Ya	Tida	Tida	Ya	Tidak
Amazon Route 53 Domain	Ya	Tida	Tida	Tida	Tida	Tidak
Cluster Pemulihan Amazon Route 53	Ya	Ya	Tida	Tida	Ya	Tidak
Config Kontrol Pemulihan Amazon Route 53	Ya	Ya	Tida	Ya	Ya	Tidak
Kesiapan Pemulihan Amazon Route 53	Ya	Ya	Tida	Ya	Ya	Ya
Amazon Route 53 Resolver	Ya	Ya	Tida	Ya	Ya	Ya
Amazon S3 Ekspres	Ya	Ya	Tida	Tida	Ya	Tidak
Gletser Amazon S3	Ya	Ya	Ya	Ya	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensia l sementara	Peran terkait layanan
Amazon SageMaker	Ya	Ya	Tida	Ya	Ya	Sebagian (Info)
Kemampuan SageMaker geospasial Amazon	Ya	Ya	Tida	Ya	Ya	Tidak
Amazon SageMaker Ground Truth Sintetis	Ya	Tida	Tida	Tida	Ya	Tidak
AWS Savings Plans	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Secrets Manager	Ya	Ya	Ya	Ya	Ya	Tidak
AWS Security Hub	Ya	Ya	Tida	Ya	Ya	Ya
Danau Keamanan Amazon	Ya	Ya	Tida	Tida	Ya	Ya

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensia I sementara	Peran terkait layanan
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
Service Quotas	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS Shield	Ya	Ya	Tidak	Ya	Ya	Ya
AWS Signer	Ya	Ya	Ya	Ya	Ya	Tidak
Amazon SimpleDB	Ya	Ya	Tidak	Tidak	Ya	Tidak
Layanan Email Sederhana Amazon (Amazon SES) v2	Ya	Sebagian (Info)	Ya	Ya	Sebagian (Info)	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
Layanan Pemberitahuan Sederhana Amazon (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
AWS SimSpacePenun	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS Site-to-Site VPN	Ya	Ya	Tidak	Tidak	Ya	Ya

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
AWS Snowball	 Ya	 Tida	 Tida	 Tida	 Ya	 Tidak
AWS Snowball Tepi	 Ya	 Tida	 Tida	 Tida	 Ya	 Tidak
AWS Snow Device Management	 Ya	 Ya	 Tida	 Ya	 Ya	 Tidak
AWS SQL Workbench	 Ya	 Ya	 Tida	 Ya	 Ya	 Tidak
AWS Step Functions	 Ya	 Ya	 Tida	 Ya	 Ya	 Tidak
AWS Storage Gateway	 Ya	 Ya	 Tida	 Ya	 Ya	 Tidak
Rantai Pasokan AWS	 Ya	 Ya	 Tida	 Ya	 Ya	 Tidak
AWS Support App in Slack	 Ya	 Tida	 Tida	 Tida	 Ya	 Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensia l sementara	Peran terkait layanan
AWS Support	Ya	Tida	Tida	Tida	Ya	Ya
AWS Support Rencana	Ya	Tida	Tida	Tida	Ya	Tidak
AWS Keberlanjutan	Ya	Tida	Tida	Tida	Ya	Tidak
AWS Systems Manager	Ya	Ya	Tida	Ya	Ya	Ya
AWS Systems Manager untuk SAP	Ya	Ya	Tida	Ya	Ya	Tidak
AWS Systems Manager GUI Connect	Ya	Tida	Tida	Tida	Ya	Tidak
AWS Systems Manager Incident Manager	Ya	Ya	Ya	Ya	Ya	Ya
AWS Systems Manager Incident Manager Kontak	Ya	Ya	Ya	Tida	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
Penyunting Tag	Ya	Tida	Tida	Tida	Ya	Tidak
AWS Pengaturan Pajak	Ya	Tida	Tida	Tida	Ya	Tidak
AWS Pembangun Jaringan Telekomunikasi	Ya	Ya	Tida	Ya	Ya	Tidak
Amazon Texttract	Ya	Tida	Tida	Tida	Ya	Tidak
Amazon Timestream	Ya	Ya	Tida	Ya	Ya	Tidak
Influxdb Amazon Timestream	Ya	Ya	Tida	Ya	Ya	Ya
AWS Tiros API (untuk Reachability Analyzer)	Ya	Tida	Tida	Tida	Tida	Tidak
Amazon Transcribe	Ya	Ya	Tida	Ya	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
AWS Transfer Family	Ya	Ya	Tidak	Ya	Ya	Tidak
Amazon Translate	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS Trusted Advisor	Sebagai (Info)	Ya	Tidak	Tidak	Sebagai	Ya
AWS Pemberitahuan Pengguna	Ya	Ya	Tidak	Ya	Ya	Ya
AWS Kontak Pemberitahuan Pengguna	Ya	Ya	Tidak	Ya	Ya	Tidak
Akses Terverifikasi AWS	Ya	Tidak	Tidak	Tidak	Ya	Tidak
Izin Terverifikasi Amazon	Ya	Ya	Tidak	Tidak	Ya	Tidak

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensia l sementara	Peran terkait layanan
Amazon Virtual Private Cloud (Amazon VPC)	Ya	Sebagian (Info)	Sebagian (Info)	Ya	Ya	Sebagian (Info)
Kisi VPC Amazon	Ya	Ya	Tidak	Ya	Ya	Tidak
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

Layanan	Tindakan	Izin tingkat sumber daya	Kebijakan berbasis sumber daya	ABAC	Kredensial sementara	Peran terkait layanan
Amazon WorkDocs	Ya	Tida	Tida	Tida	Ya	Tida
Amazon WorkMail	Ya	Ya	Tida	Ya	Ya	Ya
Alur WorkMail Pesan Amazon	Ya	Ya	Tida	Tida	Ya	Tida
Amazon WorkSpaces	Ya	Ya	Tida	Ya	Ya	Tida
Klien WorkSpaces Tipis Amazon	Ya	Ya	Tida	Ya	Ya	Tida
WorkSpaces Web Amazon	Ya	Ya	Tida	Ya	Ya	Ya
AWS X-Ray	Ya	Sebagai (Info)	Tida	Sebagai (Info)	Ya	Tida

Informasi lain

Amazon CloudFront

CloudFront tidak memiliki peran terkait layanan, tetapi Lambda @Edge memilikinya. Untuk informasi selengkapnya, lihat [Peran Tertaut Layanan untuk Lambda @Edge di Panduan Pengembang Amazon CloudFront](#).

AWS CloudTrail

CloudTrail mendukung kebijakan berbasis sumber daya hanya pada CloudTrail saluran yang digunakan untuk [integrasi CloudTrail Lake dengan](#) sumber acara di luar AWS.

CloudTrail mendukung kontrol akses berbasis tag untuk penyimpanan dan saluran data acara CloudTrail Lake. CloudTrail tidak mendukung kontrol akses berbasis tag untuk jalur.

Amazon CloudWatch

CloudWatch peran terkait layanan tidak dapat dibuat menggunakan AWS Management Console, 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 Data Multi-Wilayah Multi-Akun dan bagian AWS Config Aturan pada AWS Config Panduan 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 [Memberikan izin pengguna untuk berganti peran](#).

IAM mendukung kontrol akses berbasis tag untuk sebagian besar sumber daya IAM. Untuk informasi selengkapnya, lihat [Menandai sumber daya IAM](#).

Hanya beberapa tindakan API untuk IAM yang dapat dipanggil dengan kredensial sementara. Untuk informasi selengkapnya, lihat [Membandingkan opsi API Anda](#).

AWS IoT

Perangkat yang terhubung ke AWS IoT diautentikasi dengan menggunakan sertifikat X.509 atau menggunakan Identitas Amazon Cognito. Anda dapat melampirkan AWS IoT kebijakan ke sertifikat X.509 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 tindakan API yang menggunakan fungsi Lambda sebagai sumber daya yang diperlukan. Lapisan, pemetaan sumber peristiwa, dan sumber konfigurasi penandatanganan kode tidak didukung.

Lambda tidak memiliki peran terkait layanan, tetapi Lambda @Edge memilikinya. Untuk informasi selengkapnya, lihat [Peran Tertaut Layanan untuk Lambda @Edge di Panduan Pengembang Amazon CloudFront](#)

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 kluster ke kluster MSK Amazon yang telah dikonfigurasi untuk konektivitas [multi-VPC](#).

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 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

Kebijakan berbasis sumber daya 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

Peran terkait layanan saat ini tersedia untuk pekerjaan SageMaker Studio dan SageMaker pelatihan.

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 kredensial 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 `ses:SendEmail` atau `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 hanya untuk sumber daya objek.

Amazon S3 mendukung peran terkait layanan untuk Lensa Penyimpanan Amazon S3.

AWS Trusted Advisor

Akses API ke Trusted Advisor adalah melalui AWS Support API dan dikendalikan oleh kebijakan AWS Support IAM.

Amazon Virtual Private Cloud

Dalam kebijakan pengguna IAM, Anda tidak dapat membatasi izin ke titik akhir VPC Amazon tertentu. Semua elemen Action yang mencakup `ec2:*VpcEndpoint*` atau `ec2:DescribePrefixLists` tindakan API harus menentukan `""Resource"": "*" ""`. Untuk informasi selengkapnya, lihat [Identitas dan manajemen akses untuk titik akhir VPC dan layanan titik akhir VPC](#) di Panduan.AWS PrivateLink

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

Amazon VPC tidak memiliki peran terkait layanan, tetapi tidak. AWS Transit Gateway Untuk informasi selengkapnya, lihat [Menggunakan peran terkait layanan untuk gateway transit](#) di Panduan VPC Amazon. AWS Transit Gateway

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.

Menandatangani permintaan AWS API

Important

Jika Anda menggunakan alat AWS SDK (lihat [Contoh Kode dan Pustaka](#)) atau baris AWS perintah (CLI) untuk mengirim permintaan API AWS, Anda dapat melewati bagian ini karena klien SDK dan CLI mengautentikasi 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.

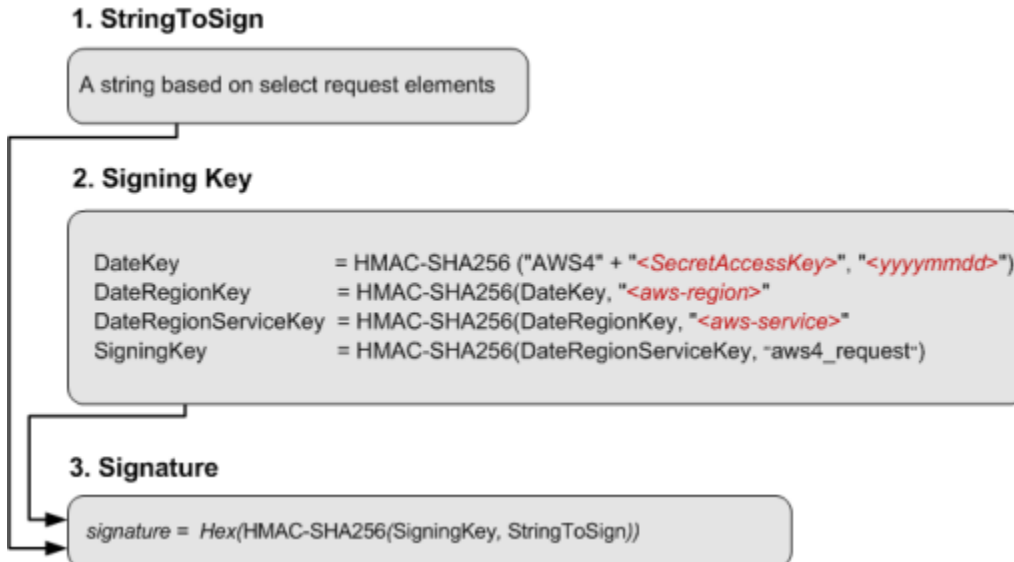
Di Wilayah yang mendukung beberapa versi tanda tangan, permintaan penandatanganan secara manual berarti Anda harus menentukan versi tanda tangan yang digunakan. Saat Anda menyediakan permintaan ke Titik Akses Multi-Wilayah, SDK dan CLI secara otomatis beralih menggunakan Signature Version 4A tanpa konfigurasi tambahan.

Informasi otentikasi yang Anda kirim dalam permintaan harus menyertakan tanda tangan. Untuk menghitung tanda tangan, pertama-tama Anda menggabungkan elemen permintaan pilih untuk membentuk string, yang disebut sebagai string untuk ditandatangani. Anda kemudian menggunakan kunci penandatanganan untuk menghitung kode otentikasi pesan berbasis hash (HMAC) dari string yang akan ditandatangani.

Di AWS Signature Version 4, Anda tidak menggunakan kunci akses rahasia untuk menandatangani permintaan. Sebagai gantinya, pertama-tama Anda menggunakan kunci akses rahasia Anda untuk mendapatkan kunci penandatanganan. Kunci penandatanganan turunan khusus untuk tanggal, layanan, dan Wilayah. Untuk informasi selengkapnya tentang cara mendapatkan kunci penandatanganan dalam bahasa pemrograman yang berbeda, lihat [Minta contoh tanda tangan](#).

Signature Version 4 adalah protokol AWS penandatanganan. AWS juga mendukung ekstensi, Signature Version 4A, yang mendukung tanda tangan untuk permintaan API Multi-wilayah. Untuk informasi lebih lanjut, lihat [a-signing-examples](#) proyek [sigv4](#) di GitHub

Diagram berikut menggambarkan proses umum komputasi tanda tangan.



- 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 yang bervariasi untuk membuat string yang akan ditandatangani. Untuk permintaan HTTP POST, POST kebijakan dalam permintaan adalah string yang Anda tandatangi.
- Untuk kunci penandatanganan, diagram menunjukkan serangkaian perhitungan, di mana hasil dari setiap langkah yang Anda masukkan ke langkah berikutnya. Langkah terakhir adalah kunci penandatanganan.
- Ketika sebuah 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. Kalau tidak, itu menolak permintaan.

Daftar Isi

- [Kapan menandatangani permintaan](#)
- [Mengapa permintaan ditandatangani](#)
- [Elemen tanda tangan permintaan AWS API](#)
- [Metode otentikasi](#)

- [Membuat permintaan AWS API yang ditandatangani](#)
- [Minta contoh tanda tangan](#)
- [Memecahkan masalah permintaan yang ditandatangani untuk API AWS](#)

Kapan menandatangani permintaan

Saat Anda menulis kode kustom yang mengirimkan permintaan API 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 memerlukan kontrol penuh atas bagaimana permintaan dikirim AWS.

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 kredensial keamanan sementara, perhitungan tanda tangan juga memerlukan token keamanan. Untuk informasi selengkapnya, lihat akses [terprogram kredensial AWS keamanan](#).

- Lindungi data dalam perjalanan

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 tersebut.

- Lindungi dari potensi serangan replay

Dalam kebanyakan kasus, permintaan harus mencapai AWS dalam waktu lima menit dari cap waktu dalam permintaan. Kalau tidak, AWS menolak permintaan itu.

Elemen tanda tangan permintaan AWS 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 Signature Version 4 dapat menjadi tugas yang rumit, dan kami menyarankan Anda menggunakan AWS SDK atau CLI bila memungkinkan.

Setiap permintaan HTTP/HTTPS yang menggunakan penandatanganan Signature Version 4 harus berisi elemen-elemen ini.

Elemen

- [Spesifikasi titik akhir](#)
- [Tindakan](#)
- [Parameter tindakan](#)
- [Tanggal](#)
- [Informasi otentikasi](#)

Spesifikasi titik akhir

Menentukan nama DNS dari titik akhir yang Anda kirim permintaan. Nama ini biasanya berisi kode layanan dan Wilayah. Misalnya, titik akhir untuk Amazon DynamoDB di Wilayah us-east-1 adalah `dynamodb.us-east-1.amazonaws.com`

Untuk permintaan HTTP/1.1, Anda harus menyertakan header. Host Untuk permintaan HTTP/2, Anda dapat menyertakan `:authority` header atau header. Host Hanya gunakan header `:authority` untuk kepatuhan dengan spesifikasi HTTP/2. Tidak semua layanan mendukung permintaan HTTP/2.

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 `CreateTable` atau tindakan Amazon EC2. `DescribeInstances`

Untuk tindakan yang didukung oleh setiap layanan, lihat [Referensi Otorisasi Layanan](#).

Parameter tindakan

Menentukan parameter untuk tindakan yang ditentukan dalam permintaan. Setiap tindakan AWS API memiliki seperangkat parameter wajib dan opsional. Versi API biasanya merupakan parameter yang diperlukan.

Untuk parameter yang didukung oleh tindakan API, lihat [Referensi API](#) untuk layanan.

Tanggal

Menentukan tanggal dan waktu permintaan. Menyertakan tanggal dan waktu dalam permintaan membantu mencegah pihak ketiga menegat permintaan Anda dan mengirimkannya kembali nanti. Tanggal yang Anda tentukan dalam lingkup 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 `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** — Gunakan AWS 4-HMAC-SHA256 untuk menentukan Signature Version 4 dengan algoritma HMAC-SHA256 hash.
- **Credential** — String yang terdiri dari ID kunci akses Anda, tanggal dalam format YYYYMMDD, kode Region, kode layanan, dan string `aws4_request` terminasi, dipisahkan oleh garis miring (/). Kode Region, kode layanan, dan string terminasi harus menggunakan karakter huruf kecil.

```
AKIAIOSFODNN7EXAMPLE/YYYYMMDD/region/service/aws4_request
```

- **Header yang ditandatangani** - Header HTTP untuk disertakan dalam tanda tangan, dipisahkan oleh titik koma (;). Misalnya, `host;x-amz-date`.
- **Signature** — String yang dikodekan heksadesimal yang mewakili tanda tangan yang dihitung. Anda harus menghitung tanda tangan menggunakan algoritme yang Anda tentukan di parameter `Algorithm`.

Metode otentikasi

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 Signature Version 4 dapat menjadi tugas yang rumit, dan kami menyarankan Anda menggunakan AWS SDK atau CLI bila memungkinkan.

Anda dapat mengekspresikan informasi otentikasi dengan menggunakan salah satu metode berikut.

Header otorisasi HTTP

AuthorizationHeader HTTP adalah metode yang paling umum untuk mengautentikasi permintaan. Semua operasi REST API (kecuali untuk unggahan berbasis browser menggunakan POST permintaan) memerlukan header ini. Untuk informasi selengkapnya tentang nilai header otorisasi, dan cara menghitung tanda tangan dan opsi terkait, lihat [Mengautentikasi Permintaan: Menggunakan Header Otorisasi \(Versi AWS Tanda Tangan 4\)](#) di Referensi API Amazon S3.

Berikut ini adalah contoh dari nilai Authorization header. Jeda 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.

```
Authorization: AWS 4-HMAC-SHA256
Credential=AKIAIOSFODNN7EXAMPLE/20130524/us-east-1/s3/aws4_request,
SignedHeaders=host;range;x-amz-date,
Signature=fe5f80f77d5fa3beca038a248ff027d0445342fe2855ddc963176630326f1024
```

Tabel berikut menjelaskan berbagai komponen nilai header Otorisasi pada contoh sebelumnya:

Komponen	Deskripsi
Otorisasi	Algoritma yang digunakan untuk menghitung tanda tangan. Anda harus memberikan nilai ini ketika Anda menggunakan AWS Signature Version 4 untuk otentikasi. String menentukan AWS Signature Version 4 (AWS 4) dan algoritma penandatanganan (). HMAC-SHA256

Komponen	Deskripsi
Kredensi	<p>ID kunci akses Anda dan informasi ruang lingkup, yang mencakup tanggal, Wilayah, dan layanan yang digunakan untuk menghitung tanda tangan.</p> <p>String ini memiliki bentuk berikut:</p> <pre><your-access-key-id>/<date>/ <aws-region>/<aws-service>/ aws4_request</pre> <p>Dimana: <date> nilai ditentukan menggunakan format YYYYMMDD. <aws-service> nilainya adalah s3 saat mengirim permintaan ke Amazon S3.</p>
SignedHeaders	<p>Daftar header permintaan yang dipisahkan titik koma yang Anda gunakan untuk menghitung <code>Signature</code>. Daftar ini hanya mencakup nama header, dan nama header harus dalam huruf kecil. Misalnya: <code>host; jangkauan; x-amz-date</code></p>
Tanda tangan	<p>Tanda tangan 256-bit dinyatakan sebagai 64 karakter heksadesimal huruf kecil. Sebagai contoh: <code>fe5f80f77d5fa3beca038a248ff027d0445342fe2855ddc963176630326f1024</code></p> <p>Perhatikan bahwa perhitungan tanda tangan bervariasi tergantung pada opsi yang Anda pilih untuk mentransfer muatan.</p>

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 presigned. Anda dapat menggunakan URL yang telah ditetapkan sebelumnya untuk menyematkan tautan yang dapat diklik dalam HTML, yang dapat berlaku hingga tujuh hari. Untuk informasi selengkapnya, lihat [Mengautentikasi Permintaan: Menggunakan Parameter Kueri \(Versi AWS Tanda Tangan 4\)](#) di Referensi API Amazon S3.

Berikut ini adalah contoh URL presigned. Jeda baris ditambahkan ke contoh ini untuk keterbacaan:

```
https://s3.amazonaws.com/examplebucket/test.txt ?
X-Amz-Algorithm=AWS4-HMAC-SHA256 &
X-Amz-Credential=<your-access-key-id>/20130721/us-east-1/s3/aws4_request &
X-Amz-Date=20130721T201207Z &
X-Amz-Expires=86400 &
X-Amz-SignedHeaders=host &X-Amz-Signature=<signature-value>
```

Note

X-Amz-CredentialNilai dalam URL menunjukkan karakter “/” hanya untuk keterbacaan.

Dalam prakteknya, itu harus dikodekan sebagai% 2F. Sebagai contoh:

```
&X-Amz-Credential=<your-access-key-id>%2F20130721%2Fus-
east-1%2Fs3%2Faws4_request
```

Tabel berikut menjelaskan parameter kueri dalam URL yang menyediakan informasi otentikasi.

Nama parameter string kueri	Deskripsi
X-Amz-Algorithm	Mengidentifikasi versi AWS Signature dan algoritma yang Anda gunakan untuk menghitung tanda tangan. Untuk AWS Signature Version 4, Anda menetapkan nilai parameter ini ke. <code>AWS4-HMAC-SHA256</code> String ini mengidentifikasi AWS Signature Version 4 (AWS 4) dan algoritma HMAC-SHA256 (). HMAC-SHA256
X-Amz-Credential	Selain ID kunci akses Anda, parameter ini juga menyediakan ruang lingkup (AWS Wilayah dan layanan) yang tanda tangannya valid. Nilai

Nama parameter string kueri	Deskripsi
	<p>ini harus sesuai dengan cakupan yang Anda gunakan dalam perhitungan tanda tangan, dibahas di bagian berikut.</p> <p>Bentuk umum untuk nilai parameter ini adalah sebagai berikut:</p> <pre><your-access-key-id>/<date>/<AWS Region>/<AWS-service>/aws4_request</pre> <p>Sebagai contoh: AKIAIOSFODNN7EXAMPLE/20130721/us-east-1/s3/aws4_request</p> <p>Untuk daftar string AWS regional, lihat Titik Akhir Regional di Referensi AWS Umum.</p>
X-Amz-Date	<p>Format tanggal dan waktu harus mengikuti standar ISO 8601, dan harus diformat dengan format. yyyyMMddTHH:mm:ssZ Misalnya jika tanggal dan waktu adalah "08/01/2016 15:32:41.982-700" maka pertama-tama harus dikonversi ke UTC (Coordinated Universal Time) dan kemudian diserahkan sebagai "20160801T223241Z".</p>

Nama parameter string kueri	Deskripsi
X-Amz-Kedaluwarsa	<p>Menyediakan periode waktu, dalam hitungan detik, di mana URL presigned yang dihasilkan 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 ditentukan sebelumnya dapat berlaku selama maksimal tujuh hari karena kunci penandatanganan yang Anda gunakan dalam perhitungan tanda tangan berlaku hingga tujuh hari.</p>
X-Amz- SignedHeaders	<p>Daftar header yang Anda gunakan untuk menghitung tanda tangan. Header berikut diperlukan dalam perhitungan tanda tangan:</p> <ul style="list-style-type: none">• Header host HTTP.• Setiap header x-amz-* yang Anda rencanakan untuk ditambahkan ke permintaan. <p>Untuk keamanan tambahan, Anda harus menandatangani semua header permintaan yang Anda rencanakan untuk disertakan dalam permintaan Anda.</p>
X-Amz-Signature	<p>Memberikan tanda tangan untuk mengautentikasi permintaan Anda. Tanda tangan ini harus sesuai dengan tanda tangan yang dihitung oleh layanan; jika tidak, layanan menolak permintaan tersebut. Misalnya, 733255ef022bec3f2a8701cd61d4b371f3f28c9f193a1f02279211d48d5193d7</p> <p>Perhitungan tanda tangan dijelaskan di bagian berikut.</p>

Nama parameter string kueri	Deskripsi
X-Amz-Security-Token	Parameter kredensial opsional jika menggunakan kredensial yang bersumber dari layanan STS.

Membuat permintaan AWS API yang ditandatangani

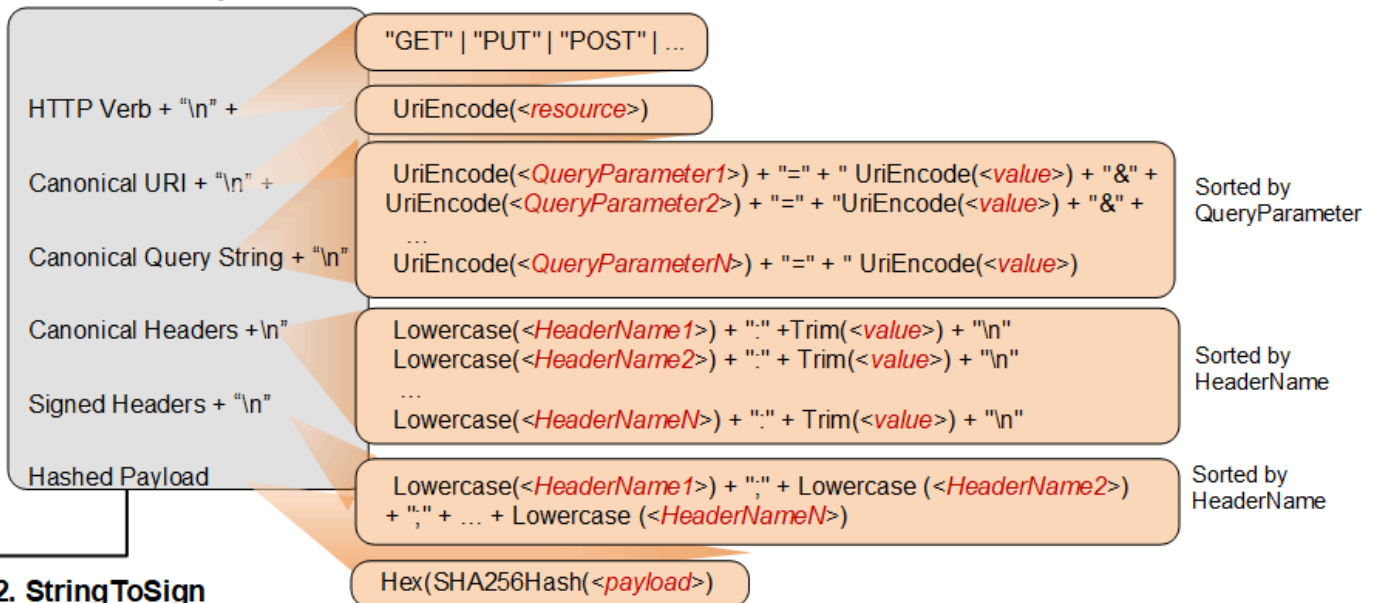
Important

Jika Anda menggunakan alat AWS SDK (lihat [Contoh Kode dan Pustaka](#)) atau baris AWS perintah (CLI) untuk mengirim permintaan API AWS, Anda dapat melewati bagian ini karena klien SDK dan CLI mengautentikasi 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.

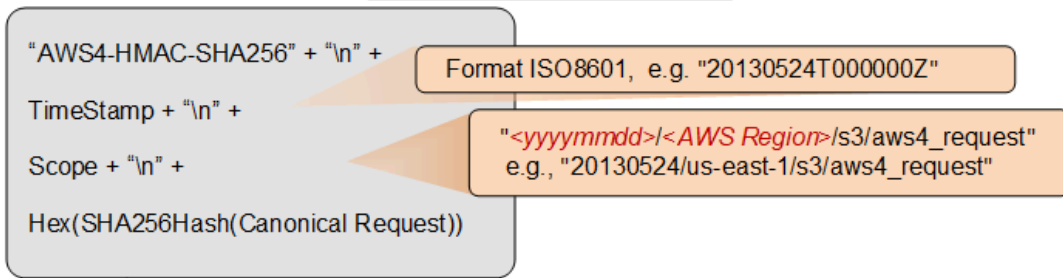
Di Wilayah yang mendukung beberapa versi tanda tangan, permintaan penandatanganan secara manual berarti Anda harus menentukan versi tanda tangan yang digunakan. Saat Anda menyediakan permintaan ke Titik Akses Multi-Wilayah, SDK dan CLI secara otomatis beralih menggunakan Signature Version 4A tanpa konfigurasi tambahan.

Berikut ini adalah ikhtisar proses untuk membuat permintaan yang ditandatangani. Untuk menghitung tanda tangan, pertama-tama Anda memerlukan string untuk ditandatangani. Anda kemudian menghitung hash HMAC-SHA256 dari string yang akan ditandatangani dengan menggunakan kunci penandatanganan. Diagram berikut menggambarkan proses, termasuk berbagai komponen string yang Anda buat untuk ditandatangani.

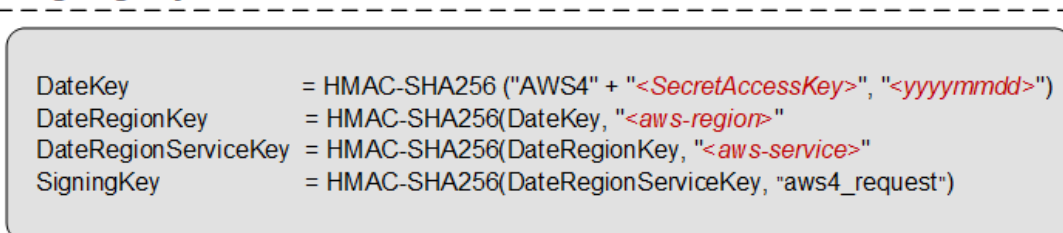
1. Canonical Request



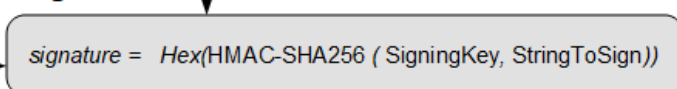
2. StringToSign



3. Signing Key




4. Signature



Tabel berikut menjelaskan fungsi yang ditunjukkan dalam diagram. Anda perlu menerapkan kode untuk fungsi-fungsi ini. Untuk informasi selengkapnya, lihat [contoh kode di AWS SDK](#).

Fungsi	Deskripsi
<code>Lowercase()</code>	Ubah string menjadi huruf kecil.
<code>Hex()</code>	Pengkodean basis 16 huruf kecil.
<code>SHA256Hash()</code>	Fungsi hash kriptografi Secure Hash Algorithm (SHA).
<code>HMAC-SHA256()</code>	Menghitung HMAC dengan menggunakan algoritma SHA256 dengan kunci penandaan yang disediakan. Ini adalah tanda tangan terakhir.
<code>Trim()</code>	Hapus spasi putih di depan atau belakang.
<code>UriEncode()</code>	<p>URI menyandikan setiap byte. <code>UriEncode()</code> harus menegakkan aturan berikut:</p> <ul style="list-style-type: none">• URI menyandikan setiap byte kecuali karakter tanpa syarat: 'A'-'Z', 'a'-'z', '0'-'9', '-', '.', '_', dan '~'.• Karakter spasi adalah karakter cadangan dan harus dikodekan sebagai "% 20" (dan bukan sebagai "+").• Setiap byte yang dikodekan URI dibentuk oleh '%' dan nilai heksadesimal dua digit dari byte.• Huruf dalam nilai heksadesimal harus huruf besar, misalnya "% 1A".• Encode karakter garis miring maju, '/', di mana-mana kecuali dalam nama kunci objek. Misalnya, jika nama kunci objek adalah <code>photos/Jan/sample.jpg</code>, garis miring ke depan dalam nama kunci tidak dikodekan.

Fungsi	Deskripsi
	<p> Important</p> <p>UriEncode Fungsi standar yang disediakan oleh platform pengembangan 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.</p> <p>Untuk melihat contoh UriEncode fungsi di Java, lihat Java Utilities di GitHub situs web.</p>

Note

Saat menandatangani permintaan, Anda dapat menggunakan AWS Signature Version 4 atau AWS Signature Version 4A. Perbedaan utama antara keduanya ditentukan oleh bagaimana tanda tangan dihitung. Dengan AWS Signature Version 4A, tanda tangan tidak menyertakan informasi khusus Wilayah dan dihitung menggunakan algoritme. AWS 4-ECDSA-P256-SHA256

Kredensial keamanan sementara

Alih-alih menggunakan kredensyal jangka panjang untuk menandatangani permintaan, Anda dapat menggunakan kredensyal keamanan sementara yang disediakan oleh (). AWS Security Token Service AWS STS

Saat Anda menggunakan kredensyal keamanan sementara, Anda harus menambahkan X-Amz-Security-Token ke header Otorisasi atau string kueri untuk menahan 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 detailnya.

Ringkasan langkah-langkah penandatanganan

Langkah 1: Buat permintaan kanonik

Atur konten permintaan Anda (host, action, header, dll.) Ke dalam format kanonik standar. Permintaan kanonik adalah salah satu input yang digunakan untuk membuat string untuk ditandatangani. Untuk detailnya, lihat [Elemen tanda tangan permintaan AWS API](#).

Langkah 2: Buat hash dari permintaan kanonik

Dapatkan kunci penandatanganan dengan melakukan serangkaian operasi hash kunci (operasi HMAC) pada tanggal permintaan, Wilayah, dan layanan, dengan kunci akses AWS rahasia Anda sebagai kunci untuk operasi hashing awal.

Langkah 3: Buat String untuk Masuk

Buat string yang akan ditandatangani dengan permintaan kanonik dan informasi tambahan seperti algoritme, tanggal permintaan, cakupan kredensial, dan intisari (hash) dari permintaan kanonik.

Langkah 4: Hitung tanda tangan

Setelah Anda mendapatkan kunci penandatanganan, Anda kemudian menghitung tanda tangan dengan melakukan operasi hash yang dikunci pada string yang akan ditandatangani. Gunakan kunci penandatanganan turunan sebagai kunci hash untuk operasi ini.

Langkah 5: Tambahkan tanda tangan ke permintaan

Setelah Anda menghitung tanda tangan, tambahkan ke header HTTP atau ke string kueri permintaan.

Langkah 1: Buat permintaan kanonik

Buat permintaan kanonik dengan menggabungkan string berikut, dipisahkan oleh karakter baris baru. Ini membantu memastikan bahwa tanda tangan yang Anda hitung dan tanda tangan yang AWS menghitung dapat cocok.

```
<HTTPMethod>\n<CanonicalURI>\n<CanonicalQueryString>\n
```



```
<CanonicalHeaders>\n
<SignedHeaders>\n
<HashedPayload>
```

- **HttpMethod** — Metode HTTP, seperti GET, PUT, HEAD, dan DELETE
- **CanonicalUri** — Versi URI yang dikodekan URI dari URI komponen jalur absolut, dimulai dengan "/" yang mengikuti nama domain dan sampai 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, /examplebucket/myphoto.jpg, adalah jalur absolut dan Anda tidak menyandikan "/" di jalur absolut:

```
http://s3.amazonaws.com/examplebucket/myphoto.jpg
```

- **CanonicalQueryString** — Parameter string kueri yang dikodekan URI. Anda URI-encode setiap nama dan nilai satu per satu. Anda juga harus mengurutkan parameter dalam string kueri kanonik menurut abjad dengan nama kunci. Penyortiran terjadi setelah pengkodean. String query dalam contoh URI berikut adalah:

```
http://s3.amazonaws.com/examplebucket?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 kueri yang sesuai akan menjadi string kosong (""). Misalnya, URI berikut mengidentifikasi ACL subresource pada bucket: examplebucket

```
http://s3.amazonaws.com/examplebucket?acl
```

CanonicalQueryString Dalam hal ini adalah sebagai berikut:

```
UriEncode("acl") + "=" + ""
```

Jika URI tidak menyertakan '?', tidak ada string kueri dalam permintaan, dan Anda mengatur string kueri kanonik ke string kosong (""). Anda masih perlu memasukkan "\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:

- hostHeader 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 kredensial keamanan sementara, Anda harus memasukkan x-amz-security-token dalam permintaan Anda. Anda harus menambahkan header ini dalam daftar **CanonicalHeaders**.

Note

x-amz-content-sha256Header diperlukan untuk permintaan Amazon S3 AWS . 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 ruang depan atau belakang.
- mengubah ruang berurutan menjadi satu ruang.

- pisahkan nilai untuk header multi-nilai menggunakan koma.
- Anda harus menyertakan header host (HTTP/1.1) atau:authority header (HTTP/2), dan header apa pun dalam tanda tangan. `x-amz-*` 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:e3b0c44298fc1c149afb4c8996fb92427ae41e4649b934ca495991b7852b855
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 semua header dalam perhitungan tanda tangan.

- ***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` Misalnya, untuk contoh sebelumnya, nilai ***SignedHeaders*** akan menjadi sebagai berikut:

```
host;x-amz-content-sha256;x-amz-date
```

- ***HashedPayload***— String yang dibuat menggunakan payload di badan permintaan HTTP sebagai masukan 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 sebagai berikut:

```
Hex(SHA256Hash(""))
```

Hash mengembalikan nilai berikut:

```
e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
```

Misalnya, saat Anda mengunggah objek dengan menggunakan permintaan PUT, Anda memberikan data objek di badan. Ketika Anda mengambil objek dengan menggunakan GET permintaan, Anda menghitung hash string kosong.

Langkah 2: Buat hash dari permintaan kanonik

Buat hash (digest) dari permintaan kanonik menggunakan algoritma yang sama yang Anda gunakan untuk membuat hash dari payload. Hash dari permintaan kanonik adalah string karakter heksadesimal huruf kecil.

Langkah 3: Buat string untuk ditandatangani

Buat string dengan menggabungkan string berikut, dipisahkan oleh karakter baris baru. Jangan akhiri string ini dengan karakter baris baru.

```
Algorithm \n
RequestDateTime \n
CredentialScope \n
HashedCanonicalRequest
```

- *Algoritma* — Algoritma yang digunakan untuk membuat hash dari permintaan kanonik. Untuk SHA-256, algoritmanya adalah. AWS4-HMAC-SHA256
- *RequestDateTime*— Tanggal dan waktu yang digunakan dalam lingkup kredensi. Nilai ini adalah waktu UTC saat ini dalam format ISO 8601 (misalnya,). 20130524T000000Z
- *CredentialScope*— Ruang lingkup kredensi. Ini membatasi tanda tangan yang dihasilkan ke Wilayah dan layanan yang ditentukan. *String memiliki format berikut: YYYYMMDD/region/service /aws4_request.*
- *HashedCanonicalRequest*— Hash dari permintaan kanonik. Nilai ini dihitung pada Langkah 2.

Berikut ini adalah contoh string untuk menandatangani.

```
"AWS4-HMAC-SHA256" + "\n" +  
timeStampISO8601Format + "\n" +  
<Scope> + "\n" +  
Hex(SHA256Hash(<CanonicalRequest>))
```

Langkah 4: Hitung tanda tangan

Di Versi AWS Tanda Tangan 4, alih-alih menggunakan kunci AWS akses untuk menandatangani permintaan, Anda membuat kunci penandatanganan yang dicakup ke Wilayah dan layanan tertentu sebagai informasi otentikasi yang akan Anda tambahkan ke permintaan Anda.

```
DateKey = HMAC-SHA256("AWS4"+"<SecretAccessKey>", "<YYYYMMDD>")  
DateRegionKey = HMAC-SHA256(<DateKey>, "<aws-region>")  
DateRegionServiceKey = HMAC-SHA256(<DateRegionKey>, "<aws-service>")  
SigningKey = HMAC-SHA256(<DateRegionServiceKey>, "aws4_request")
```

Untuk daftar string Region, lihat [Titik Akhir Regional](#) di Referensi AWS Umum.

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.

Input yang dibutuhkan

- Sebuah string, *Key*, yang berisi kunci akses rahasia Anda
- Sebuah string, *Date*, yang berisi tanggal yang digunakan dalam lingkup kredensi, dalam format YYYYMMDD
- Sebuah string *Region*, yang berisi kode Wilayah (misalnya, *us-east-1*)
- Sebuah string *Service*, yang berisi kode layanan (misalnya, *ec2*)
- String untuk menandatangani bahwa Anda membuat pada langkah sebelumnya.

Untuk menghitung tanda tangan

1. Menggabungkan "AWS4" dan kunci akses rahasia. Panggil fungsi hash dengan string gabungan sebagai kunci dan string tanggal sebagai data.

```
kDate = hash("AWS4" + Key, Date)
```

2. Panggil fungsi hash dengan hasil panggilan sebelumnya sebagai kunci dan string Region sebagai data.

```
kRegion = hash(kDate, Region)
```

3. Panggil fungsi hash dengan hasil panggilan sebelumnya sebagai kunci dan string layanan sebagai data.

```
kService = hash(kRegion, Service)
```

4. Panggil fungsi hash dengan hasil panggilan sebelumnya sebagai kunci dan "aws4_request" sebagai data.

```
kSigning = hash(kService, "aws4_request")
```

5. Panggil fungsi hash dengan hasil panggilan sebelumnya sebagai kunci dan string untuk ditandatangani sebagai data. Hasilnya adalah tanda tangan sebagai nilai biner.

```
signature = hash(kSigning, string-to-sign)
```

6. Mengkonversi tanda tangan dari biner ke representasi heksadesimal, dalam karakter huruf kecil.

Langkah 5: Tambahkan tanda tangan ke permintaan

Example Contoh: Header otorisasi

Contoh berikut menunjukkan Authorization header untuk DescribeInstances tindakan. 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
```

Example Contoh: Permintaan dengan parameter otentikasi dalam string kueri

Contoh berikut menunjukkan kueri untuk `DescribeInstances` tindakan 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
```

Kode sumber di AWS SDK

AWS SDK menyertakan kode sumber GitHub untuk menandatangani permintaan AWS API. Untuk contoh kode, lihat [Contoh proyek dalam AWS repositori sampel](#)

- AWS SDK for .NET — [AWS4Signer.cs](#)
- AWS SDK for C++ — [AWSAuthV4Signer.cpp](#)
- AWS SDK for Go — [v4.go](#)
- AWS SDK for Java — [BaseAws4Signer.java](#)
- AWS SDK for JavaScript - [v4.js](#)
- AWS SDK for PHP — [SignatureV4.php](#)
- AWS SDK for Python (Boto) - [signers.py](#)
- AWS SDK for Ruby — [penandatanganan.rb](#)

Minta contoh tanda tangan

Contoh permintaan AWS penandatanganan berikut menunjukkan kepada Anda bagaimana Anda dapat menggunakan SigV4 untuk menandatangani permintaan yang dikirim tanpa AWS SDK atau alat AWS baris perintah.

Unggahan Amazon S3 berbasis browser menggunakan HTTP POST

[Permintaan Autentikasi: Unggahan Berbasis Browser](#) menjelaskan tanda tangan dan informasi relevan yang digunakan Amazon S3 untuk menghitung tanda tangan setelah menerima permintaan.

[Contoh: Unggahan Berbasis Browser menggunakan HTTP POST \(Menggunakan AWS Tanda Tangan Versi 4\)](#) memberikan informasi lebih lanjut dengan contoh kebijakan POST dan formulir yang dapat Anda gunakan untuk mengunggah file. Contoh kebijakan dan kredensial fiktif menunjukkan alur kerja dan tanda tangan yang dihasilkan serta hash kebijakan.

Permintaan yang diautentikasi VPC Lattice

[Contoh untuk permintaan terautentikasi Signature Version 4 \(SigV4\)](#) memberikan contoh Python dan Java yang menunjukkan bagaimana Anda dapat melakukan penandatanganan permintaan dengan dan tanpa pencegat khusus.

Menggunakan Signature Versi 4 dengan Amazon Translate

[Menggunakan Signature Version 4 dengan Amazon Translate](#) menunjukkan cara menggunakan program Python untuk menambahkan informasi otentikasi ke permintaan Amazon Translate. Contoh membuat permintaan POST, membuat struktur JSON yang berisi teks yang akan diterjemahkan dalam isi (payload) permintaan, dan meneruskan informasi otentikasi di header Otorisasi.

Menggunakan Signature Versi 4 dengan Neptune

[Contoh: Menghubungkan ke Neptune Menggunakan Python dengan Signature Version 4 Signature](#) Menandatangani menunjukkan cara membuat permintaan yang ditandatangani ke Neptune menggunakan Python. Contoh ini mencakup variasi untuk menggunakan kunci akses atau kredensial sementara.

Menandatangani permintaan HTTP ke S3 Glacier

[Contoh Perhitungan Tanda Tangan untuk Streaming API](#) memandu Anda melalui detail pembuatan tanda tangan untuk Unggah Arsip (arsip POST), salah satu dari dua API streaming di S3 Glacier.

Membuat Permintaan HTTP ke Amazon SWF

[Membuat Permintaan HTTP ke Amazon SWF](#) menunjukkan konten header untuk permintaan JSON ke Amazon SWF.

Perhitungan tanda tangan untuk streaming API di Amazon OpenSearch Service

[Menandatangani permintaan pencarian Amazon OpenSearch Service dengan AWS SDK for PHP Versi 3](#) mencakup contoh cara mengirim permintaan HTTP yang ditandatangani ke OpenSearch Amazon Service.

Contoh proyek dalam AWS repositori sampel

Contoh proyek berikut menunjukkan cara menandatangani permintaan untuk membuat permintaan Rest API ke AWS layanan dengan bahasa umum seperti Python, Node.js, Java, C #, Go dan Rust.

Proyek Signature Version 4a

[a-signing-examples](#) Proyek [sigv4](#) memberikan contoh cara menandatangani permintaan dengan Sigv4a untuk membuat permintaan Rest API Layanan AWS dengan bahasa umum seperti Python, Node.js, Java, C #, Go dan Rust.

[Membuat permintaan menggunakan Multi-Region Access Point \(MRAP\)](#) menggunakan Signature Version 4a untuk mengakses data di Amazon S3 menggunakan Python boto 3.

Publikasikan ke AWS IoT Core

[Kode Python untuk dipublikasikan AWS IoT Core menggunakan protokol HTTPS](#) memberikan panduan tentang cara mempublikasikan pesan untuk AWS IoT Core menggunakan protokol Https dan autentikasi AWS SigV4. Ini memiliki dua implementasi referensi - satu di Python dan lainnya di NodeJs

[Aplikasi .Net Framework untuk mempublikasikan AWS IoT Core menggunakan protokol HTTPS](#) memberikan panduan tentang cara mempublikasikan pesan untuk AWS IoT Core menggunakan protokol Https dan autentikasi AWS SigV4. Proyek ini juga mencakup implementasi setara inti .NET.

Memecahkan masalah permintaan yang ditandatangani untuk API AWS

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 rumit, dan kami menyarankan 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 Layanan AWS dihitung. `Unauthorized` Kesalahan 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 autentikasi IAM.
- Kredensi IAM yang digunakan untuk menandatangani permintaan tidak benar 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 AWS Signature versi 2 (SigV2) ke AWS Signature versi 4 (SigV4) sebelum menjelajahi solusi kesalahan lainnya. Services, seperti Amazon S3, dan Wilayah tidak lagi mendukung penandatanganan SigV2.

Kemungkinan penyebab

- [Kesalahan kredensi](#)
- [Permintaan kanonik dan kesalahan string penandatanganan](#)
- [Kesalahan cakupan kredensi](#)
- [Kesalahan penandatanganan kunci](#)

Kesalahan kredensi

Pastikan bahwa permintaan API ditandatangani dengan SigV4. Jika permintaan API tidak ditandatangani, Anda mungkin menerima kesalahan: `Missing Authentication Token`. [Tambahkan tanda tangan yang hilang](#) dan kirim ulang permintaan.

Verifikasi bahwa kredensi 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 berwenang untuk membuat permintaan. Untuk detailnya, lihat [Memecahkan masalah akses ditolak pesan kesalahan](#).

Permintaan kanonik dan kesalahan string penandatanganan

Jika Anda salah menghitung permintaan kanonik di [Langkah 2: Buat hash dari permintaan kanonik](#) atau [Langkah 3: Buat 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, itu menghitung ulang tanda tangan. Jika ada perbedaan nilai, maka tanda tangan tidak cocok. Bandingkan permintaan dan string kanonik dengan permintaan Anda 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
----- A list of signed
content-type;host;x-amz-date 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. AWS

Header permintaan API

Pastikan bahwa header otorisasi SigV4 yang Anda tambahkan [Langkah 4: Hitung tanda tangan](#) menyertakan kunci kredensial yang benar mirip dengan yang berikut ini:

```
Authorization: AWS4-HMAC-SHA256
Credential=AKIAIOSFODNN7EXAMPLE/20130524/us-east-1/s3/aws4_request,
SignedHeaders=host;range;x-amz-date,
Signature=example-generated-signature
```

Jika kunci kredensial hilang atau salah, Anda mungkin menerima kesalahan: `Authorization header requires 'Credential' parameter.` `Authorization header requires 'Signature' parameter.` Pastikan bahwa permintaan otorisasi SigV4 juga menyertakan tanggal permintaan menggunakan salah satu atau HTTP Date header. `x-amz-date`

Kesalahan cakupan kredensial

Cakupan kredensial yang Anda buat [Langkah 3: 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 kredensial.

Tanggal

Jika cakupan kredensial tidak menentukan tanggal yang sama dengan `x-amz-date` header, langkah verifikasi tanda tangan gagal dengan pesan galat 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 galat berikut:

```
Signature not yet current: date is still later than date
```

Jika permintaan telah kedaluwarsa, langkah verifikasi tanda tangan gagal dengan pesan galat berikut:

```
Signature expired: date is now earlier than date
```

Wilayah

Jika cakupan kredensi tidak menentukan Wilayah yang sama dengan permintaan, langkah verifikasi tanda tangan gagal dengan pesan galat 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 galat berikut:

```
Credential should be scoped to correct service: 'service'
```

String penghentian

Jika cakupan kredensi tidak berakhir `aws4_request`, langkah verifikasi tanda tangan gagal dengan pesan galat 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 Anda 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. 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 kebijakan](#) Bagian ini menjelaskan AWS permintaan, bagaimana mereka diautentikasi, dan bagaimana 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 memetakan 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 global](#)— Bagian ini mencakup daftar semua kunci kondisi AWS global yang dapat Anda gunakan untuk membatasi izin dalam kebijakan IAM.
- [Kunci konteks IAM dan AWS STS kondisi](#)— Bagian 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 [Memvalidasi kebijakan IAM](#). Untuk mempelajari selengkapnya tentang pemeriksaan kebijakan IAM Access Analyzer dan rekomendasi yang dapat ditindaklanjuti, lihat validasi kebijakan [IAM Access Analyzer](#).

Topik

- [Elemen IAM JSON:Version](#)
- [Elemen kebijakan IAM JSON: Id](#)
- [Elemen kebijakan IAM JSON:Statement](#)
- [Elemen kebijakan IAM JSON: Sid](#)
- [Elemen kebijakan IAM JSON:Effect](#)
- [AWS Elemen kebijakan JSON: Principal](#)
- [Elemen kebijakan AWS JSON: NotPrincipal](#)

- [Elemen kebijakan IAMAction](#)
- [Elemen kebijakan IAM JSON: NotAction](#)
- [Elemen kebijakan IAM JSON: Resource](#)
- [Elemen kebijakan IAM JSON:NotResource](#)
- [Elemen kebijakan IAM JSON: Condition](#)
- [Elemen kebijakan IAM: Variabel dan tag](#)
- [Elemen kebijakan IAM JSON: Tipe data yang didukung](#)

Elemen 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 `VersionStatement` elemen berikut di semua kebijakan Anda.

```
{
  "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.

```
{
  "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 beranda” Anda sendiri di konsol Amazon S3.) Kebijakan ini mencakup variabel `aws:username`, yang diganti selama evaluasi kebijakan dengan nama pengguna dari permintaan. Untuk informasi selengkapnya, lihat [Pengantar](#).

```
{
  "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:::BUCKET-NAME",
      "Condition": {"StringLike": {"s3:prefix": [
        "",
        "home/",
        "home/${aws:username}/"
      ]}}
    },
    {
      "Effect": "Allow",
      "Action": "s3:*",
      "Resource": [
        "arn:aws:s3:::BUCKET-NAME/home/${aws:username}",
        "arn:aws:s3:::BUCKET-NAME/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.

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

SidElemen 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:Effect

Elemen Effect diperlukan dan menentukan apakah pernyataan tersebut mengakibatkan diizinkan atau ditolak. Nilai yang valid untuk Effect adalah Allow dan Deny. EffectNilai ini memeka huruf 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`. Untuk informasi selengkapnya, lihat [Logika evaluasi kebijakan](#).

AWS Elemen kebijakan JSON: Principal

Gunakan `Principal` elemen dalam kebijakan JSON berbasis sumber daya untuk menentukan prinsipal 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 tepercaya) 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 file. AWS KMS key

Anda tidak dapat menggunakan `Principal` elemen dalam kebijakan berbasis identitas. Kebijakan berbasis identitas adalah kebijakan izin yang Anda lampirkan ke identitas IAM (pengguna, grup, atau peran). Dalam kasus tersebut, kepala sekolah secara implisit adalah identitas di mana kebijakan dilampirkan.

Topik

- [Menentukan prinsipal](#)
- [Akun AWS kepala sekolah](#)
- [Kepala peran IAM](#)
- [Kepala sesi peran](#)
- [Prinsip pengguna IAM](#)
- [Kepala Pusat Identitas IAM](#)
- [AWS STS prinsip sesi pengguna federasi](#)
- [AWS prinsipal layanan](#)
- [AWS prinsip layanan di Wilayah keikutsertaan](#)
- [Semua kepala sekolah](#)
- [Informasi lain](#)

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
- Sesi pengguna federasi
- AWS layanan
- Semua kepala sekolah

Anda tidak dapat mengidentifikasi grup pengguna sebagai prinsipal dalam kebijakan (seperti kebijakan berbasis sumber daya) karena grup terkait dengan izin, bukan autentikasi, 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. Saat Anda

menentukan lebih dari satu prinsipal dalam suatu elemen, Anda memberikan izin kepada setiap prinsipal. Ini logis OR dan bukan logisAND, karena Anda mengautentikasi sebagai satu prinsipal pada satu waktu. Jika Anda menyertakan lebih dari satu nilai, gunakan tanda kurung siku ([dan]) dan comma-delimit setiap entri untuk array. Contoh kebijakan berikut mendefinisikan izin untuk akun 123456789012 atau akun 555555555555.

```
"Principal" : {
  "AWS": [
    "123456789012",
    "555555555555"
  ]
}
```

Note

Anda tidak dapat menggunakan wildcard untuk mencocokkan sebagian nama utama atau ARN.

Akun AWS kepala sekolah

Anda dapat menentukan Akun AWS pengenal dalam `Principal` elemen kebijakan berbasis sumber daya atau dalam kunci kondisi yang mendukung prinsipal. Ini mendelegasikan wewenang 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::account-id:root`), atau formulir 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" }
```

Akun ARN 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 dapat mengubahnya menjadi ARN utama. Ini tidak mengubah fungsionalitas kebijakan.

Beberapa AWS layanan mendukung opsi tambahan untuk menentukan pokok akun. Misalnya, Amazon S3 memungkinkan Anda menentukan [ID pengguna kanonik](#) 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"  
}
```

Kepala 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 di AWS atau pengguna dari penyedia identitas eksternal (iDP). Ketika prinsipal atau identitas mengambil peran, mereka menerima kredensial keamanan sementara dengan izin peran yang diasumsikan. Ketika mereka menggunakan kredensial sesi tersebut untuk melakukan operasi AWS, mereka menjadi kepala sesi peran.

Peran IAM adalah identitas yang ada di IAM. Peran mempercayai identitas lain yang diautentikasi, seperti prinsipal di AWS atau pengguna dari penyedia identitas eksternal. Ketika kepala sekolah atau identitas mengambil peran, mereka menerima kredensial keamanan sementara. Mereka kemudian

dapat menggunakan kredensial tersebut sebagai prinsipal sesi peran untuk melakukan operasi di AWS

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, lihat [Logika 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, ARN tersebut akan berubah menjadi ID utama unik peran 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. Hasil akhirnya 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 saat Anda menyimpan kebijakan.

Atau, Anda dapat menentukan prinsipal peran sebagai prinsipal dalam kebijakan berbasis sumber daya atau [membuat kebijakan izin luas yang menggunakan](#) kunci kondisi `aws:PrincipalArn`. Saat Anda menggunakan kunci ini, prinsipal sesi peran diberikan izin berdasarkan ARN peran yang diasumsikan, dan bukan ARN dari sesi yang dihasilkan. Karena AWS tidak mengonversi ARN kunci kondisi menjadi ID, izin yang diberikan ke ARN peran tetap ada jika Anda menghapus peran dan kemudian membuat peran baru dengan nama yang sama. Jenis kebijakan berbasis identitas, 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.

Kepala sesi peran

Anda dapat menentukan sesi peran dalam `Principal` elemen kebijakan berbasis sumber daya atau dalam kunci kondisi yang mendukung prinsipal. Ketika prinsipal atau identitas mengambil peran, mereka menerima kredensial keamanan sementara dengan izin peran yang diasumsikan. Ketika mereka menggunakan kredensi sesi tersebut untuk melakukan operasi AWS, mereka menjadi kepala sesi peran.

Format yang Anda gunakan untuk prinsipal sesi peran bergantung pada AWS STS operasi yang digunakan untuk mengambil peran.

Selain itu, administrator dapat merancang proses untuk mengontrol bagaimana sesi peran dikeluarkan. Misalnya, mereka dapat memberikan solusi satu-klik untuk penggunaanya yang membuat nama sesi yang dapat diprediksi. Jika administrator melakukan ini, Anda dapat menggunakan prinsipal sesi peran dalam kebijakan atau kunci kondisi. Jika tidak, Anda dapat menentukan peran ARN sebagai prinsipal dalam kunci `aws:PrincipalArn` kondisi. Cara Anda menentukan peran sebagai prinsipal dapat mengubah izin efektif untuk sesi yang dihasilkan. Untuk informasi selengkapnya, lihat [Kepala peran IAM](#).

Prinsip sesi peran yang diasumsikan

Prinsipal sesi peran yang diasumsikan adalah prinsip sesi yang dihasilkan dari penggunaan operasi AWS STS `AssumeRole`. Untuk informasi lebih lanjut tentang prinsipal mana yang dapat mengambil peran menggunakan operasi ini, lihat [Membandingkan operasi AWS STS API](#)

Untuk menentukan ARN sesi peran yang diasumsikan dalam elemen, gunakan format `Principal` berikut:

```
"Principal": { "AWS": "arn:aws:sts::AWS-account-ID:assumed-role/role-name/role-session-name" }
```

Saat Anda menentukan sesi peran yang diasumsikan dalam `Principal` elemen, Anda tidak dapat menggunakan wildcard "*" yang berarti semua sesi. Prinsipal harus selalu menamai sesi tertentu.

Kepala sekolah sesi OIDC

Prinsipal sesi OIDC adalah prinsipal sesi yang dihasilkan dari penggunaan operasi AWS STS `AssumeRoleWithWebIdentity`. Anda dapat menggunakan penyedia OIDC eksternal (iDP) untuk

masuk, dan kemudian mengambil peran IAM menggunakan operasi ini. Ini memanfaatkan federasi identitas dan mengeluarkan sesi peran. Untuk informasi lebih lanjut tentang prinsipal mana yang dapat mengambil peran menggunakan operasi ini, lihat [Membandingkan operasi AWS STS API](#)

Saat Anda mengeluarkan peran dari penyedia OIDC, Anda mendapatkan jenis prinsipal sesi khusus ini yang mencakup informasi tentang penyedia OIDC.

Gunakan tipe utama ini dalam kebijakan Anda untuk mengizinkan atau menolak akses berdasarkan penyedia identitas web tepercaya. Untuk menentukan ARN sesi peran OIDC dalam `Principal` elemen kebijakan kepercayaan peran, gunakan format berikut:

```
"Principal": { "Federated": "cognito-identity.amazonaws.com" }
```

```
"Principal": { "Federated": "www.amazon.com" }
```

```
"Principal": { "Federated": "graph.facebook.com" }
```

```
"Principal": { "Federated": "accounts.google.com" }
```

Prinsipal sesi SAMP

Prinsipal sesi SAMP adalah prinsipal sesi yang dihasilkan dari penggunaan AWS STS `AssumeRoleWithSAML` operasi. Anda dapat menggunakan penyedia identitas SAMP eksternal (iDP) untuk masuk, dan kemudian mengambil peran IAM menggunakan operasi ini. Ini memanfaatkan federasi identitas dan mengeluarkan sesi peran. Untuk informasi lebih lanjut tentang prinsipal mana yang dapat mengambil peran menggunakan operasi ini, lihat [Membandingkan operasi AWS STS API](#)

Saat Anda mengeluarkan peran dari penyedia identitas SAFL, Anda mendapatkan jenis prinsipal sesi khusus ini yang mencakup informasi tentang penyedia identitas SAFL.

Gunakan tipe utama ini dalam kebijakan Anda untuk mengizinkan atau menolak akses berdasarkan penyedia identitas SAMP tepercaya. Untuk menentukan ARN sesi peran identitas SAMP 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 sebuah `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”. Prinsipal harus selalu memberi nama pengguna tertentu.

Important

Jika `Principal` elemen Anda dalam kebijakan kepercayaan peran berisi ARN yang mengarah ke pengguna IAM tertentu, maka IAM akan mengubah ARN menjadi ID utama unik pengguna saat Anda menyimpan kebijakan tersebut. 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

Dalam IAM Identity Center, prinsipal dalam kebijakan berbasis sumber daya harus didefinisikan sebagai prinsipal. Akun AWS Untuk menentukan akses, rujuk peran ARN dari izin yang ditetapkan di blok kondisi. Untuk detailnya, lihat [Mereferensikan set izin dalam kebijakan sumber daya, Amazon EKS, dan AWS KMS](#) di Panduan Pengguna Pusat Identitas IAM.

AWS STS prinsip sesi pengguna federasi

Anda dapat menentukan sesi pengguna federasi dalam `Principal` elemen kebijakan berbasis sumber daya atau dalam kunci kondisi yang mendukung prinsipal.

Important

AWS merekomendasikan agar Anda menggunakan sesi pengguna AWS STS federasi hanya jika diperlukan, seperti ketika [akses pengguna root diperlukan](#). Sebagai gantinya, [gunakan peran untuk mendelegasikan izin](#).

Prinsipal sesi pengguna AWS STS federasi adalah prinsip sesi yang dihasilkan dari penggunaan AWS STS `GetFederationToken` operasi. Dalam hal ini, AWS STS menggunakan [federasi identitas](#) sebagai metode untuk mendapatkan token akses sementara alih-alih menggunakan peran IAM.

Di AWS, pengguna IAM atau Pengguna root akun AWS dapat mengautentikasi menggunakan kunci akses jangka panjang. Untuk informasi lebih lanjut tentang kepala sekolah mana yang dapat berfederasi menggunakan operasi ini, lihat [Membandingkan operasi AWS STS API](#)

- Pengguna federasi IAM — Pengguna IAM bergabung menggunakan `GetFederationToken` operasi yang menghasilkan prinsipal sesi pengguna federasi untuk pengguna IAM tersebut.
- Pengguna root federasi — Pengguna root federasi menggunakan `GetFederationToken` operasi yang menghasilkan prinsipal sesi pengguna federasi untuk pengguna root tersebut.

Ketika pengguna IAM atau pengguna root meminta kredensial sementara dari AWS STS menggunakan operasi ini, mereka memulai sesi pengguna gabungan sementara. ARN sesi ini didasarkan pada identitas asli yang difederasi.

Untuk menentukan ARN sesi pengguna federasi 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. Prinsipal layanan adalah pengidentifikasi untuk suatu 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 prinsip 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": "*"` .

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 membuka [AWS layanan yang bekerja dengan IAM](#), memeriksa apakah layanan memiliki Ya di kolom Peran yang dikaitkan dengan layanan, dan membuka tautan Ya untuk melihat dokumentasi peran yang dikaitkan dengan layanan untuk layanan tersebut. Temukan bagian Izin Peran Ditautkan Layanan untuk layanan tersebut dapat 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 prinsip layanan di Wilayah keikutsertaan

Anda dapat meluncurkan sumber daya di beberapa AWS Wilayah dan beberapa Wilayah yang harus Anda pilih. Untuk daftar lengkap Wilayah yang harus Anda pilih, lihat [Mengelola AWS Wilayah](#) di Referensi Umum AWS panduan.

Ketika suatu AWS layanan di Wilayah keikutsertaan 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 suatu AWS layanan di Wilayah keikutsertaan 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 keikutsertaan. `ap-east-1` Anda ingin mengonfigurasi notifikasi bucket S3 untuk mempublikasikan 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-regionalisasi dari prinsipal layanan `s3.amazonaws.com`, dalam kebijakan akses topik, `sns:Publish` permintaan dari bucket ke topik akan gagal. Contoh berikut menentukan prinsip layanan S3 non-regionalisasi dalam elemen `Principal` kebijakan kebijakan akses topik SNS.

```
"Principal": { "Service": "s3.amazonaws.com" }
```

Karena bucket terletak di Wilayah keikutsertaan 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 ketika AWS layanan di Wilayah 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 prinsipal layanan S3 regional dalam elemen `Principal` kebijakan akses topik SNS.

```
"Principal": { "Service": "s3.ap-east-1.amazonaws.com" }
```

Kebijakan sumber daya atau daftar izin berbasis prinsip layanan untuk permintaan Lintas wilayah dari Wilayah keikutsertaan ke Wilayah lain hanya akan berhasil jika Anda menentukan nama utama layanan regional.

Note

Untuk kebijakan kepercayaan peran IAM, sebaiknya gunakan nama utama layanan non-regional. 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. [Kebijakan berbasis sumber daya](#) izin 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 berniat untuk memberikan akses publik atau anonim. Jika tidak, tentukan prinsip, layanan, atau AWS akun yang dimaksudkan dalam `Principal` elemen dan kemudian batasi akses lebih lanjut dalam elemen. `Condition` Hal ini terutama berlaku untuk kebijakan kepercayaan peran IAM, karena mereka 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 ada izin lain yang diperlukan. Untuk prinsipal di akun lain, mereka juga harus memiliki izin berbasis identitas di akun mereka yang memungkinkan mereka mengakses sumber daya Anda. Ini disebut [akses lintas akun](#).

Untuk pengguna anonim, elemen berikut ini setara:

```
"Principal": "*" 
```

```
"Principal" : { "AWS" : "*" } 
```

Anda tidak dapat menggunakan wildcard untuk mencocokkan sebagian nama utama atau ARN.

Contoh berikut menunjukkan kebijakan berbasis sumber daya yang dapat digunakan alih-alih [Menentukan dengan *NotPrincipalDeny*](#) untuk secara eksplisit menolak semua prinsip kecuali yang ditentukan dalam elemen. Condition

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "UsePrincipalArnInsteadOfNotPrincipalWithDeny",
      "Effect": "Deny",
      "Action": "s3:*",
      "Principal": "*",
      "Resource": [
        "arn:aws:s3::BUCKETNAME/*",
        "arn:aws:s3::BUCKETNAME"
      ],
      "Condition": {
        "ArnNotEquals": {
          "aws:PrincipalArn": "arn:aws:iam::444455556666:user/user-name"
        }
      }
    }
  ]
}
```

Informasi lain

Untuk informasi selengkapnya, lihat hal berikut:

- [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 AWS Key Management Service Pengembang
- [Pengidentifikasi akun](#) di Referensi Umum AWS
- [Federasi OIDC](#)

Elemen kebijakan AWS JSON: NotPrincipal

Anda dapat menggunakan `NotPrincipal` elemen untuk menolak akses ke semua prinsipal kecuali pengguna IAM, pengguna federasi, peran IAM,, AWS layanan Akun AWS, atau prinsip lain yang ditentukan dalam elemen. `NotPrincipal`

Anda dapat menggunakannya 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.

`NotPrincipal` harus digunakan dengan `"Effect": "Deny"`. Menggunakannya `"Effect": "Allow"` dengan tidak didukung.

Important

Sangat sedikit skenario yang membutuhkan penggunaan `NotPrincipal`. Kami menyarankan Anda menjelajahi opsi otorisasi lain sebelum Anda memutuskan untuk menggunakannya `NotPrincipal`. 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. Untuk informasi selengkapnya, lihat [Semua kepala sekolah](#).

Menentukan dengan `NotPrincipalDeny`

Saat Anda menggunakan `NotPrincipal` dengan `Deny`, Anda juga harus menyebutkan 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, lalu peran tersebut, dan kemudian pengguna peran yang diasumsikan. 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.

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 prinsip IAM apa pun 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, lihat [Batas izin untuk entitas IAM](#).

Note

Sebagai praktik terbaik, Anda harus memasukkan ARN untuk akun dalam kebijakan Anda. Beberapa layanan memerlukan ARN akun, meskipun ini tidak diperlukan dalam semua kasus. Kebijakan yang sudah ada mana pun tanpa ARN yang diperlukan akan terus bekerja, tetapi kebijakan baru yang mencakup layanan ini harus memenuhi persyaratan ini. IAM tidak melacak layanan ini, dan oleh karena itu merekomendasikan Anda untuk selalu menyertakan akun ARN.

Contoh berikut menunjukkan cara menggunakan `NotPrincipal` dan "Effect": "Deny" dalam pernyataan kebijakan yang sama dengan efektif.

Example Contoh pengguna IAM di akun yang sama atau berbeda

Dalam contoh berikut, semua prinsipal kecuali pengguna bernama Bob di Akun AWS 444455556666 secara eksplisit ditolak akses ke sumber daya. Perhatikan bahwa sebagai praktik terbaik, `NotPrincipal` elemen berisi ARN dari pengguna Bob dan Bob milik `()arn:aws:iam::444455556666:root`. Akun AWS Jika `NotPrincipal` elemen hanya berisi ARN Bob, efek kebijakan mungkin secara eksplisit menolak akses ke Akun AWS yang berisi pengguna Bob. Dalam beberapa kasus, pengguna tidak dapat memiliki izin lebih banyak daripada

akun induknya, jadi jika akun Bob secara eksplisit ditolak maka Bob mungkin tidak dapat mengakses sumber daya tersebut.

Contoh ini berfungsi sebagaimana dimaksud ketika merupakan bagian dari pernyataan kebijakan dalam kebijakan berbasis sumber daya yang dilampirkan ke sumber daya yang sama atau berbeda Akun AWS (bukan 444455556666). Contoh ini tidak memberikan akses kepada Bob, namun hanya menghapus Bob dari daftar prinsipal yang secara eksplisit ditolak. Agar Bob dapat mengakses sumber daya, pernyataan kebijakan lainnya harus secara eksplisit mengizinkan akses menggunakan "Effect": "Allow".

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Deny",
    "NotPrincipal": {"AWS": [
      "arn:aws:iam::444455556666:user/Bob",
      "arn:aws:iam::444455556666:root"
    ]},
    "Action": "s3:*",
    "Resource": [
      "arn:aws:s3:::BUCKETNAME",
      "arn:aws:s3:::BUCKETNAME/*"
    ]
  }]
}
```

Example Contoh peran IAM dalam akun yang sama atau berbeda

Dalam contoh berikut, semua prinsipal kecuali pengguna peran yang dianggap bernama Akun AWS 444455556666 secara eksplisit ditolak akses ke cross-account-audit-app sumber daya. Sebagai praktik terbaik, `NotPrincipal` elemen berisi ARN dari pengguna peran yang diasumsikan (), peran (cross-account-read-only-peran-cross-account-audit-app), dan peran yang Akun AWS dimiliki (444455556666). Jika elemen `NotPrincipal` ini tidak memiliki ARN peran tersebut, efek kebijakan mungkin secara eksplisit menolak akses ke peran tersebut. Demikian pula, jika `NotPrincipal` elemen tersebut kehilangan ARN dari peran Akun AWS tersebut, efek dari kebijakan tersebut mungkin secara eksplisit menolak akses ke Akun AWS dan semua entitas dalam akun itu. Dalam beberapa kasus, pengguna peran yang diasumsikan tidak dapat memiliki izin lebih dari peran induknya, dan peran tidak dapat memiliki izin lebih dari induknya Akun AWS, jadi ketika peran atau akun secara eksplisit ditolak aksesnya, pengguna peran yang diasumsikan mungkin tidak dapat mengakses sumber daya.

Contoh ini berfungsi sebagaimana dimaksud ketika merupakan bagian dari pernyataan kebijakan dalam kebijakan berbasis sumber daya yang dilampirkan ke sumber daya yang berbeda Akun AWS (bukan 444455556666). Contoh ini dengan sendirinya tidak mengizinkan akses ke pengguna peran yang diasumsikan `cross-account-audit-app`, itu hanya menghilangkan `cross-account-audit-app` dari daftar kepala sekolah yang secara eksplisit ditolak. Untuk memberikan `cross-account-audit-app` akses ke sumber daya, pernyataan kebijakan lain harus secara eksplisit mengizinkan penggunaan akses. "Effect": "Allow"

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Deny",
    "NotPrincipal": {"AWS": [
      "arn:aws:sts::444455556666:assumed-role/cross-account-read-only-role/cross-account-audit-app",
      "arn:aws:iam::444455556666:role/cross-account-read-only-role",
      "arn:aws:iam::444455556666:root"
    ]},
    "Action": "s3:*",
    "Resource": [
      "arn:aws:s3:::Bucket_AccountAudit",
      "arn:aws:s3:::Bucket_AccountAudit/*"
    ]
  }]
}
```

Saat Anda menentukan sesi peran yang diasumsikan di elemen `NotPrincipal`, Anda tidak dapat menggunakan wildcard (*) yang berarti "semua sesi". Prinsipal harus selalu menamai sesi tertentu.

Elemen kebijakan IAMAction

Elemen `Action` menguraikan tindakan khusus atau tindakan yang akan diizinkan atau ditolak. Pernyataan harus mencakup elemen `Action` atau `NotAction`. Setiap layanan AWS memiliki rangkaian tindakannya sendiri yang menjelaskan tugas yang dapat Anda lakukan dengan layanan tersebut. Misalnya, daftar tindakan untuk Amazon S3 dapat ditemukan di [Menentukan Izin dalam Kebijakan dalam](#) Panduan Pengguna Amazon Simple Storage Service, 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.

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 SQ

```
"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 (*) untuk memberikan akses ke semua tindakan yang ditawarkan produk AWS tertentu. Misalnya, elemen `Action` berikut berlaku untuk semua tindakan S3.

```
"Action": "s3:*"
```

Anda juga dapat menggunakan wildcard (*) 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 selengkapya, 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. Hal ini tidak memungkinkan pengguna untuk menggunakan operasi API S3 ListAllMyBuckets, karena tindakan tersebut memerlukan sumber daya "*". 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 eksplisit 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.

```
{
  "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 ke AWS berdasarkan Wilayah yang diminta](#).

Elemen kebijakan IAM JSON: Resource

Elemen `Resource` menentukan objek atau objek-objek yang dicakup oleh pernyataan tersebut. Pernyataan harus mencakup elemen `Resource` atau `NotResource`. Anda menentukan sumber daya menggunakan ARN. Untuk informasi selengkapnya tentang format ARN, lihat [ARN IAM](#).

Setiap layanan memiliki serangkaian sumber daya sendiri. Meskipun Anda selalu menggunakan ARN untuk menentukan sumber daya, rincian ARN untuk sumber daya tergantung pada layanan dan sumber daya. Untuk informasi tentang cara menentukan sumber daya, lihat dokumentasi untuk layanan yang ingin Anda tulis pernyataan.

Note

Beberapa layanan tidak mengizinkan Anda menentukan tindakan untuk sumber daya individu; sebagai gantinya, tindakan apa pun yang Anda daftarkan di elemen `Action` atau `NotAction` berlaku untuk semua sumber daya dalam layanan tersebut. Dalam kasus ini, Anda menggunakan wildcard `*` di elemen `Resource`.

Contoh berikut mengacu pada antrian Amazon SQS tertentu.

```
"Resource": "arn:aws:sqs:us-east-2:account-ID-without-hyphens:queue1"
```

Contoh berikut mengacu pada pengguna IAM bernama Bob dalam file Akun AWS.

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 wildcard sebagai bagian dari sumber daya ARN. Anda dapat menggunakan karakter wildcard (* dan?) dalam segmen ARN (bagian dipisahkan oleh titik dua) untuk mewakili kombinasi karakter dengan tanda bintang (*) dan karakter tunggal dengan tanda tanya (?). Anda dapat menggunakan beberapa karakter * atau ? di setiap segmen. Jika wildcard (*) adalah karakter terakhir dari segmen ARN sumber daya, itu dapat diperluas untuk mencocokkan melampaui batas titik dua. Kami menyarankan Anda menggunakan wildcard (* dan?) dalam segmen ARN dipisahkan oleh titik dua.

Note

Anda tidak dapat menggunakan wildcard di segmen layanan yang mengidentifikasi produk. AWS Untuk informasi selengkapnya tentang segmen ARN, lihat [Amazon Resource Name \(ARN\)](#)

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:::DOC-EXAMPLE-BUCKET/*"
```

Karakter asterisk (*) dapat diperluas untuk menggantikan segala sesuatu dalam segmen, termasuk karakter seperti garis miring (/) yang mungkin tampak sebagai pembatas dalam namespace layanan tertentu. Misalnya, pertimbangkan Amazon S3 ARN berikut karena logika ekspansi wildcard yang sama berlaku untuk semua layanan.

```
"Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*/test/*"
```

Wildcard di ARN berlaku untuk semua objek berikut dalam bucket, tidak hanya objek pertama yang terdaftar.

```
DOC-EXAMPLE-BUCKET/1/test/object.jpg  
DOC-EXAMPLE-BUCKET/1/2/test/object.jpg  
DOC-EXAMPLE-BUCKET/1/2/test/3/object.jpg  
DOC-EXAMPLE-BUCKET/1/2/3/test/4/object.jpg
```

```
DOC-EXAMPLE-BUCKET/1///test///object.jpg
DOC-EXAMPLE-BUCKET/1/test/.jpg
DOC-EXAMPLE-BUCKET//test/object.jpg
DOC-EXAMPLE-BUCKET/1/test/
```

Pertimbangkan dua objek terakhir dalam daftar sebelumnya. Nama objek Amazon S3 dapat secara valid dimulai atau diakhiri dengan karakter garis miring (/) pembatas konvensional. Meskipun "/" berfungsi sebagai pembatas, tidak ada signifikansi tertentu ketika karakter ini digunakan dalam ARN sumber daya. Hal ini diperlakukan sama dengan karakter valid lainnya. ARN tidak akan cocok dengan objek berikut:

```
DOC-EXAMPLE-BUCKET/1-test/object.jpg
DOC-EXAMPLE-BUCKET/test/object.jpg
DOC-EXAMPLE-BUCKET/1/2/test.jpg
```

Menentukan beberapa sumber daya

Anda dapat menentukan beberapa sumber daya. 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 mengizinkan akses ke tabel Amazon DynamoDB yang cocok dengan nama pengguna saat ini.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "dynamodb:*",
```

```
"Resource": "arn:aws:dynamodb:us-east-2:account-id: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 `HRPayroll` seharusnya tidak diizinkan untuk mengakses sumber daya Amazon S3; apa pun kecuali `Payroll` folder dalam `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.

```
{
  "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 ke `HRBucket`, atau menambahkan sumber daya ke Amazon S3 yang seharusnya tidak boleh diakses, Anda harus menambahkan namanya ke daftar `Resource`. 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 tindakan tersebut juga mencakup tindakan Amazon EC2, maka kebijakan ini tidak akan menolak akses ke sumber daya EC2. Untuk mempelajari tindakan mana dalam layanan yang memungkinkan penetapan ARN sumber daya, lihat [Tindakan, Sumber Daya, dan Kunci Kondisi untuk Layanan AWS](#).

`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 di AWS di semua sumber daya kecuali bucket S3 HRBucket. 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 eksplisit, 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: Condition

Elemen `Condition` (atau blok `Condition`) memungkinkan Anda menentukan ketentuan saat kebijakan berlaku. Elemen `Condition` bersifat opsional. Dalam `Condition` elemen, Anda membangun ekspresi di mana Anda menggunakan [operator kondisi](#) (sama, kurang dari, dan lainnya) untuk mencocokkan kunci konteks dan nilai dalam kebijakan terhadap kunci dan nilai dalam konteks permintaan. Untuk mempelajari selengkapnya tentang konteks permintaan, lihat [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. Kunci konteks khusus layanan 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 [Kunci konteks IAM dan AWS STS kondisi](#)

Nama kunci konteks tidak peka huruf besar/kecil. Misalnya, menyertakan kunci `aws:SourceIP` konteks setara dengan pengujian untuk `AWS:SourceIp`. Sensitivitas huruf besar/huruf dari nilai kunci konteks tergantung pada [operator kondisi](#) yang Anda gunakan. Misalnya, kondisi berikut mencakup `StringEquals` operator untuk memastikan bahwa hanya permintaan yang dibuat berdasarkan `johnDoe` kecocokan. Nama pengguna `JohnDoe` ditolak aksesnya.

```
"Condition" : { "StringEquals" : { "aws:username" : "johnDoe" } }
```

Ketentuan berikut menggunakan operator [StringEqualsIgnoreCase](#) untuk mencocokkan nama pengguna `johnDoe` atau `JohnDoe`.

```
"Condition" : { "StringEqualsIgnoreCase" : { "aws:username" : "johnDoe" } }
```

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 memungkinkan Anda membuat beberapa nama kunci yang hanya berbeda 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 AWS KMS enkripsi.

- Untuk daftar semua operator kondisi dan deskripsi tentang cara kerjanya, lihat [Operator kondisi](#).
- Kecuali ditentukan lain, semua kunci konteks dapat memiliki beberapa nilai. Untuk deskripsi tentang cara menangani kunci konteks yang memiliki beberapa nilai, lihat [Kunci konteks multivaluasi](#).
- Untuk daftar semua kunci konteks yang tersedia secara global, lihat [AWS 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](#) AWS, AWS mengumpulkan informasi permintaan ke dalam konteks permintaan. Informasi ini digunakan untuk mengevaluasi dan mengotorisasi permintaan. 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](#).

Ketika permintaan dikirimkan, AWS mengevaluasi setiap kunci konteks dalam kebijakan dan mengembalikan nilai true, false, not present, dan kadang-kadang null (string data kosong). Kunci konteks yang tidak ada dalam permintaan dianggap sebagai ketidakcocokan. Misalnya, kebijakan berikut memungkinkan penghapusan perangkat autentikasi multi-faktor (MFA) Anda sendiri, tetapi hanya jika Anda telah masuk menggunakan MFA dalam satu jam terakhir (3.600 detik).

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "AllowRemoveMfaOnlyIfRecentMfa",
    "Effect": "Allow",
    "Action": [
      "iam:DeactivateMFADevice"
    ],
    "Resource": "arn:aws:iam::*:user/${aws:username}",
    "Condition": {
```

```

    "NumericLessThanEquals": {"aws:MultiFactorAuthAge": "3600"}
  }
}

```

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 CLI atau AWS API, kunci tidak ada. Dalam hal ini, kuncinya tidak ada, dan kunci tidak akan cocok.
- Null — Untuk kunci konteks yang ditentukan oleh pengguna, seperti meneruskan tag dalam permintaan, dimungkinkan untuk menyertakan string kosong. Dalam hal ini, nilai dalam konteks permintaan adalah null. Nilai null mungkin akan mengembalikan nilai tersebut dalam beberapa kasus. Misalnya, jika Anda menggunakan operator [ForAllValues](#) kondisi multivalued dengan kunci [aws:TagKeys](#) konteks, Anda dapat mengalami hasil yang tidak terduga jika konteks permintaan mengembalikan null. Untuk informasi lebih lanjut, lihat [aws: TagKeys](#) and [Kunci konteks multivaluasi](#).

Blok ketentuan

Contoh berikut menunjukkan format dasar dari elemen Condition:

```
"Condition": {"StringLike": {"s3:prefix": ["janedoe/*"]}}
```

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 `janedoe/*`. 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.

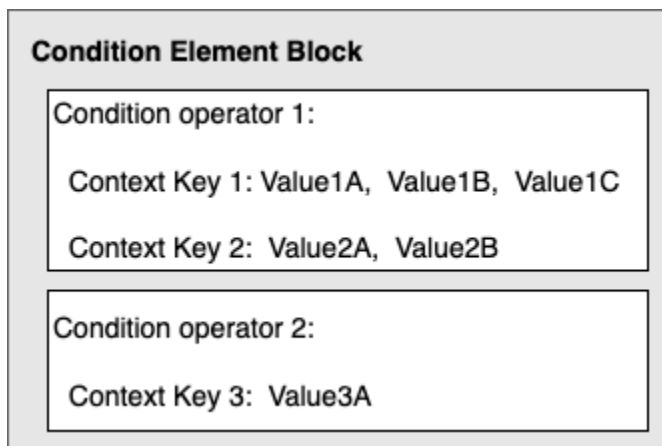
```
"Condition": {"StringLike": {"s3:prefix": ["${aws:username}/*"]}}
```

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 `dynamodb:Attributes` kunci 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 [Kunci konteks multivaluasi](#).

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.



Untuk informasi selengkapnya, lihat [Kunci konteks multivaluasi](#).

Elemen kebijakan IAM JSON: 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: Condition](#).

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 kanan tidak ada, kondisi tersebut benar. Logika ini berlaku untuk semua operator kondisi kecuali... [IfExists](#) dan [cek 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.

Operator ketentuan	Deskripsi
<code>StringEquals</code>	Kecocokan yang tepat, peka terhadap huruf besar-kecil

Operator ketentuan	Deskripsi
<code>StringNotEquals</code>	Pencocokan dinegasikan
<code>StringEqualsIgnoreCase</code>	Pencocokan yang tepat, mengabaikan huruf besar-kecil
<code>StringNotEqualsIgnoreCase</code>	Pencocokan yang dinegasikan, mengabaikan huruf besar-kecil
<code>StringLike</code>	Kecocokan kepekaan huruf besar-kecil. Nilainya dapat mencakup wildcard pencocokan multi-karakter (*) dan wildcard pencocokan karakter tunggal (?) di mana saja di string. Anda harus menentukan wildcard untuk mencapai kecocokan string paral.
	<div data-bbox="625 871 657 913" style="float: left; margin-right: 5px;"></div> <div data-bbox="673 871 747 913">Note</div> <p>Jika kunci berisi beberapa nilai, <code>StringLike</code> dapat dikualifikasikan dengan operator yang ditetapkan—<code>ForAllValues:StringLike</code> dan <code>ForAnyValue:StringLike</code>. Untuk informasi selengkapnya, lihat Kunci konteks multivaluasi.</p>
<code>StringNotLike</code>	Kecocokan dinegasikan kepekaan huruf besar-kecil. Nilainya dapat mencakup wildcard pencocokan multi-karakter (*) atau wildcard pencocokan karakter tunggal (?) di mana saja di string.

Misalnya, pernyataan berikut berisi `Condition` elemen yang menggunakan [aws:PrincipalTag](#) kunci untuk menentukan bahwa prinsipal yang membuat permintaan harus ditandai dengan kategori `iamuser-admin` pekerjaan.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:*AccessKey*",
    "Resource": "arn:aws:iam::account-id: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.

Anda dapat menggunakan [variabel kebijakan](#) dengan operator ketentuan 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.

```

{
  "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:::BUCKET-NAME",
      "Condition": {"StringLike": {"s3:prefix": [
        "",
        "home/",
        "home/${aws:username}/"
      ]}}}
  ],
  {
    "Effect": "Allow",

```

```

    "Action": "s3:*",
    "Resource": [
      "arn:aws:s3:::BUCKET-NAME/home/${aws:username}",
      "arn:aws:s3:::BUCKET-NAME/home/${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, lihat [Amazon S3: Memungkinkan pengguna Amazon Cognito mengakses objek dalam bucket mereka](#)

Pencocokan wildcard

Operator kondisi string melakukan pencocokan tanpa pola yang tidak menerapkan format yang telah ditentukan sebelumnya. ARN dan operator kondisi Tanggal adalah subset dari operator string yang menegakkan struktur pada nilai kunci kondisi. Saat Anda menggunakan StringLike atau StringNotLike operator untuk kecocokan string paralel dari ARN atau tanggal, pencocokan mengabaikan bagian struktur mana yang di-wildcarded.

Misalnya, kondisi berikut mencari kecocokan sebagian ARN menggunakan operator kondisi yang berbeda.

Ketika ArnLike digunakan, partisi, layanan, account-id, resource-type, dan sebagian resource-id bagian dari ARN harus memiliki pencocokan yang tepat dengan ARN dalam konteks permintaan. Hanya wilayah dan jalur sumber daya yang memungkinkan pencocokan sebagian.

```

"Condition": {"ArnLike": {"aws:SourceArn": "arn:aws:cloudtrail:*:111122223333:trail/*"}}

```

Ketika StringLike digunakan sebagai pengganti ArnLike, pencocokan mengabaikan struktur ARN dan memungkinkan pencocokan paralel, terlepas dari bagian yang dikartu liar.

```

"Condition": {"StringLike": {"aws:SourceArn": "arn:aws:cloudtrail:*:111122223333:trail/*"}}

```

ARN	ArnLike	StringLike
arn:aws:cloudtrail:us-west-2:111122223333:trail/keuangan	Pertandingan	Pertandingan
arn:aws:cloudtrail:us-east-2:111122223333:trail/keuangan/arsip	Pertandingan	Pertandingan
arn:aws:cloudtrail:us-east-2:444455556666:user/111122223333:trail/finance	Tidak ada kecocokan	Pertandingan

Operator ketentuan numerik

Operator ketentuan numerik memungkinkan Anda membangun elemen `Condition` yang membatasi akses berdasarkan perbandingan kunci ke nilai bilangan bulat atau desimal.

Operator ketentuan	Deskripsi
<code>NumericEquals</code>	Pencocokan
<code>NumericNotEquals</code>	Pencocokan dinegasikan
<code>NumericLessThan</code>	Pencocokan “Kurang dari”
<code>NumericLessThanEquals</code>	Pencocokan “kurang dari atau sama dengan”
<code>NumericGreaterThan</code>	Pencocokan “lebih besar dari”
<code>NumericGreaterThanEquals</code>	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 `example_bucket` sekali waktu.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:ListBucket",
    "Resource": "arn:aws:s3:::example_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.

Anda tidak dapat menggunakan [variabel kebijakan](#) dengan operator ketentuan `Numeric`.

Operator ketentuan tanggal

Operator ketentuan tanggal memungkinkan Anda membangun elemen `Condition` yang membatasi akses berdasarkan perbandingan kunci ke nilai tanggal/waktu. Anda menggunakan operator kondisi ini dengan [aws:CurrentTime](#) kunci atau [aws:EpochTime](#) kunci. Anda harus menentukan nilai tanggal/waktu dengan salah satu [Implementasi W3C dari format tanggal ISO 8601](#) atau dalam waktu epoch (UNIX).

Note

Wildcard tidak diizinkan untuk operator ketentuan tanggal.

Operator ketentuan	Deskripsi
<code>DateEquals</code>	Mencocokkan tanggal tertentu
<code>DateNotEquals</code>	Pencocokan dinegasikan

Operator ketentuan	Deskripsi
DateLessThan	Sesuai sebelum tanggal dan waktu tertentu
DateLessThanEquals	Sesuai pada atau sebelum tanggal dan waktu tertentu
DateGreaterThan	Sesuai setelah tanggal dan waktu tertentu
DateGreaterThanEquals	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.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:*AccessKey*",
    "Resource": "arn:aws:iam:*account-id: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. Kuncinya tidak ada dalam AWS CLI, AWS API, atau permintaan AWS SDK yang dibuat menggunakan kunci akses. Dalam contoh ini, jika pengguna IAM mencoba untuk melihat atau mengedit kunci akses, permintaan ditolak.


Anda tidak dapat menggunakan [variabel kebijakan](#) dengan operator ketentuan `Date`.

Operator ketentuan Boolean

Ketentuan Boolean memungkinkan Anda membangun elemen `Condition` yang membatasi akses berdasarkan perbandingan kunci "benar" atau "salah."

Operator ketentuan	Deskripsi
Bool	Pencocokan Boolean

Misalnya, kebijakan berbasis identitas ini menggunakan operator Bool kondisi dengan [aws:SecureTransport](#) kunci untuk menolak mereplikasi objek dan tag objek ke bucket tujuan dan isinya jika permintaan tidak melalui SSL.

 **Important**

Kebijakan ini tidak mengizinkan tindakan apa pun. Gunakan kebijakan ini bersama dengan kebijakan lain yang mengizinkan tindakan tertentu.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "BooleanExample",
      "Action": "s3:ReplicateObject",
      "Effect": "Deny",
      "Resource": [
        "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
        "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
      ],
      "Condition": {
        "Bool": {
          "aws:SecureTransport": "false"
        }
      }
    }
  ]
}
```

Jika kunci yang Anda sebutkan dalam ketentuan kebijakan tidak ada dalam konteks permintaan, nilainya tidak cocok. Konteks `aws:SecureTransport` permintaan mengembalikan true atau false.

Anda dapat menggunakan [variabel kebijakan](#) dengan operator ketentuan Boolean.

Operator ketentuan biner

Operator ketentuan `BinaryEquals` memungkinkan Anda membangun elemen `Condition` yang menguji nilai kunci yang berada dalam format biner. Ini membandingkan nilai dari byte kunci yang ditentukan untuk byte terhadap perwakilan terkode [base-64](#) dari nilai biner di dalam kebijakan.

```
"Condition" : {
  "BinaryEquals": {
    "key" : "QmluYXJ5VmFsdWVJbkJhc2U2NA=="
  }
}
```

Jika kunci yang Anda sebutkan dalam ketentuan kebijakan tidak ada dalam konteks permintaan, nilainya tidak cocok.

Anda tidak dapat menggunakan [variabel kebijakan](#) dengan operator ketentuan Binary.

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 standar CIDR (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 0. Untuk mempelajari apakah layanan mendukung IPv6, lihat dokumentasi untuk layanan tersebut.

Operator ketentuan	Deskripsi
<code>IpAddress</code>	Alamat IP atau rentang yang ditentukan
<code>NotIpAddress</code>	Semua alamat IP kecuali alamat dan rentang IP yang ditentukan

Misalnya, pernyataan berikut menggunakan operator ketentuan `IpAddress` dengan kunci `aws:SourceIp` untuk menentukan bahwa permintaan harus datang dari rentang IP 203.0.113.0 hingga 203.0.113.255.

```
{
```

```
"Version": "2012-10-17",
"Statement": {
  "Effect": "Allow",
  "Action": "iam:*AccessKey*",
  "Resource": "arn:aws:iam:*account-id: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.

Anda tidak dapat menggunakan [variabel kebijakan](#) dengan operator ketentuan `IpAddress`.

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 yang sudah Anda miliki untuk memastikan kebijakan terus berfungsi saat Anda melakukan transisi ke IPv6.

```
{
  "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. Ini dapat terjadi, misalnya, jika Anda menggunakan AWS CloudFormation untuk memanggil Amazon EC2 untuk membuat instance 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.

Operator ketentuan	Deskripsi
<code>ArnEquals</code> , <code>ArnLike</code>	Kecocokan huruf besar-kecil dari ARN. Masing-masing dari enam komponen ARN yang dibatasi titik dua diperiksa secara terpisah dan masing-masing dapat menyertakan wildcard pencocokan multi-karakter (*) atau wildcard pencocokan karakter tunggal (?). Operator <code>ArnEquals</code> dan <code>ArnLike</code> kondisi berperilaku identik.
<code>ArnNotEquals</code> , <code>ArnNotLike</code>	Pencocokan negasi untuk ARN. Operator <code>ArnNotEquals</code> dan <code>ArnNotLike</code> kondisi berperilaku identik.

Anda dapat menggunakan [variabel kebijakan](#) dengan operator ketentuan 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`.

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

```
"Principal": {"AWS": "123456789012"},
>Action": "SQS:SendMessage",
"Resource": "arn:aws:sqs:REGION:123456789012:QUEUE-ID",
"Condition": {"ArnEquals": {"aws:SourceArn":
"arn:aws:sns:REGION:123456789012:TOPIC-ID"}}
}
}
```

Jika kunci yang Anda sebutkan dalam ketentuan kebijakan tidak ada dalam konteks permintaan, nilainya tidak cocok. 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.

... `IfExists` operator kondisi

Anda dapat menambahkan `IfExists` ke akhir nama operator kondisi apa pun kecuali `Null` kondisi —misalnya, `StringLikeIfExists`. Anda melakukan ini untuk mengatakan "Jika kunci kebijakan ada dalam konteks permintaan, proses kuncinya 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 yang dinegasikan seperti `StringNotEqualsIfExists`, permintaan masih ditolak meskipun tag 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:

```
{
```

```

"Version": "2012-10-17",
"Statement": {
  "Sid": "THISPOLICYDOESNOTWORK",
  "Effect": "Allow",
  "Action": "ec2:RunInstances",
  "Resource": "*",
  "Condition": {"StringLike": {"ec2:InstanceType": [
    "t1.*",
    "t2.*",
    "m3.*"
  ]}}
}
}

```

Maksud dari kebijakan sebelumnya adalah untuk memungkinkan pengguna meluncurkan instance apa pun yang bertipe t1, atau t2 m3. Namun, meluncurkan instance memerlukan akses 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 `ec2:InstanceType`”, maka izinkan tindakan hanya jika nilai kunci dimulai dengan `t1.`, `t2.`, atau `m3.`. 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.

```

{
  "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": "*"
    }
}
```

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 ketentuan ini untuk menentukan apakah pengguna menggunakan kredensial mereka sendiri untuk operasi atau kredensial sementara. Jika pengguna menggunakan kredensial sementara, maka kuncinya `aws:TokenIssueTime` ada dan memiliki nilai. Contoh berikut menunjukkan ketentuan yang menyatakan bahwa pengguna tidak boleh menggunakan kredensial sementara (kunci tidak boleh ada) agar pengguna dapat menggunakan API Amazon EC2.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Action": "ec2:*",
    "Effect": "Allow",
    "Resource": "*"
  }
}
```

```
"Condition":{"Null":{"aws:TokenIssueTime":"true"}}
}
```

Kondisi dengan beberapa kunci konteks atau nilai

Anda dapat menggunakan `Condition` elemen kebijakan untuk menguji beberapa kunci konteks atau beberapa nilai untuk kunci konteks tunggal dalam permintaan. Saat Anda membuat permintaan ke AWS, baik secara terprogram maupun melalui AWS Management Console, permintaan Anda mencakup informasi tentang prinsipal, operasi, tanda, 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

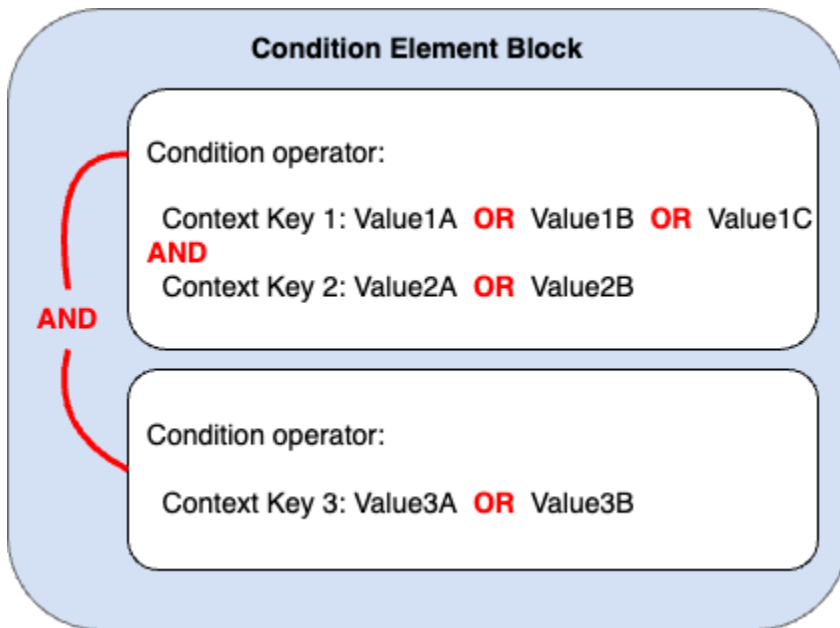
- [Logika evaluasi untuk beberapa kunci konteks atau nilai](#)
- [Logika evaluasi untuk operator kondisi pencocokan yang dinegasikan](#)

Logika evaluasi untuk beberapa kunci konteks atau nilai

SEBUAH `Condition` elemen dapat berisi beberapa operator kondisi, dan setiap operator kondisi dapat berisi beberapa konteks pasangan kunci-nilai. Kebanyakan kunci konteks mendukung menggunakan beberapa nilai, kecuali ditentukan lain.

- Jika pernyataan kebijakan Anda memiliki beberapa [operator kondisi](#), operator kondisi dievaluasi menggunakan logisAND.
- Jika pernyataan kebijakan Anda memiliki beberapa kunci konteks yang melekat pada operator kondisi tunggal, kunci konteks dievaluasi menggunakan logisAND.
- Jika operator kondisi tunggal menyertakan beberapa nilai untuk kunci konteks, nilai-nilai tersebut dievaluasi menggunakan logisOR.
- Jika operator kondisi pencocokan tunggal dinegasikan mencakup beberapa nilai untuk kunci konteks, nilai-nilai tersebut dievaluasi menggunakan logisNOR.

Semua kunci konteks dalam blok elemen kondisi harus menyelesaikan `true` untuk memanggil yang diinginkan `Allow` atau `Deny` efek. Gambar berikut menggambarkan logika evaluasi untuk kondisi dengan beberapa operator kondisi dan konteks pasangan kunci-nilai.



Misalnya, kebijakan bucket S3 berikut menggambarkan bagaimana gambar sebelumnya direpresentasikan dalam kebijakan. Blok kondisi termasuk 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 menyelesaikan benar. Pengguna yang membuat permintaan harus memiliki kedua kunci tag utama, `departemen` dan `peran`, yang mencakup salah satu nilai kunci tag yang ditentukan dalam kebijakan. Juga, ARN utama pengguna yang membuat permintaan harus sesuai dengan salah satu `aws:PrincipalArn` nilai yang ditentukan dalam kebijakan yang akan dievaluasi sebagai benar.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ExamplePolicy",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::222222222222:root"
      },
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
      "Condition": {
        "StringEquals": {
          "aws:PrincipalTag/departemen": [
            "finance",

```

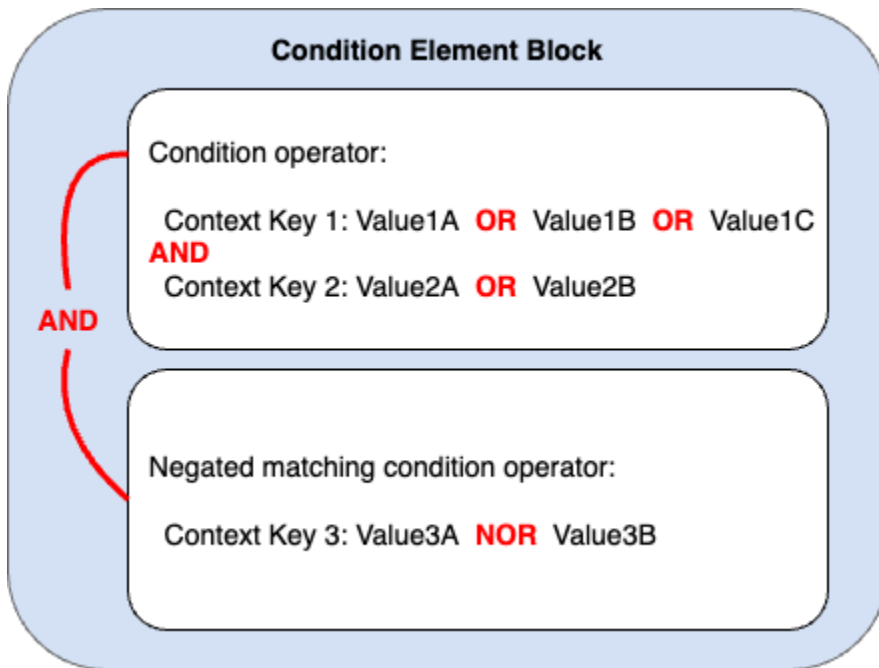


```
    "hr",
    "legal"
  ],
  "aws:PrincipalTag/role": [
    "audit",
    "security"
  ]
},
"ArnLike": {
  "aws:PrincipalArn": [
    "arn:aws:iam::222222222222:user/Ana",
    "arn:aws:iam::222222222222:user/Mary"
  ]
}
}
]
}
```

Logika evaluasi untuk operator kondisi pencocokan yang dinegasikan

Beberapa [operator kondisi](#), seperti `StringNotEquals` atau `ArnNotLike`, gunakan pencocokan yang dinegasikan untuk membandingkan pasangan kunci-nilai konteks dalam kebijakan Anda terhadap pasangan kunci-nilai konteks dalam permintaan. Ketika beberapa nilai ditentukan untuk kunci konteks tunggal dalam kebijakan dengan operator kondisi pencocokan dinegasikan, izin efektif bekerja seperti logisNOR. Dalam pencocokan dinegasikan, logisNOR atau NOT OR mengembalikan true hanya jika semua nilai mengevaluasi ke false.

Gambar berikut menggambarkan logika evaluasi untuk kondisi dengan beberapa operator kondisi dan konteks pasangan kunci-nilai. 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 termasuk operator kondisi `StringEquals` dan `ArnNotLike`, dan kunci konteks `saws:PrincipalTag` dan `saws:PrincipalArn`. Untuk memanggil yang diinginkan `Allow` atau `Deny` efek, semua kunci konteks di blok kondisi harus menyelesaikan benar. Pengguna yang membuat permintaan harus memiliki kedua kunci tag utama, `departemen` dan `peran`, yang mencakup salah satu nilai kunci tag yang ditentukan dalam kebijakan. Sejak `ArnNotLikeOperator` kondisi menggunakan pencocokan yang dinegasikan, ARN utama pengguna yang membuat permintaan tidak boleh cocok dengan salah satu `saws:PrincipalArn` nilai yang ditentukan dalam kebijakan yang akan dievaluasi sebagai benar.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ExamplePolicy",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::222222222222:root"
      },
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::DOC-EXAMPLE-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"
    ]
  }
}
```

Kunci konteks bernilai tunggal vs. multivaluasi

Perbedaan antara kunci konteks bernilai tunggal dan multivalued tergantung pada jumlah nilai dalam [konteks permintaan](#), bukan jumlah nilai dalam kondisi kebijakan.

- Kunci konteks kondisi bernilai tunggal memiliki paling banyak satu nilai dalam konteks permintaan. Misalnya, Anda dapat menandai sumber daya diAWS. Tag sumber daya disimpan sebagai pasangan nilai kunci tag. Kunci tag sumber daya dapat memiliki nilai tag tunggal. Oleh karena itu, [the section called “ResourceTag”](#) adalah kunci konteks bernilai tunggal. Jangan gunakan operator set kondisi dengan kunci konteks bernilai tunggal.
- Kunci konteks kondisi multivalued dapat memiliki beberapa nilai dalam konteks permintaan. Misalnya, Anda dapat menandai sumber daya AWS dan menyertakan beberapa pasangan nilai kunci tag dalam permintaan. Oleh karena itu, [the section called “TagKeys”](#) adalah kunci konteks multivalued. Kunci konteks multivaluasi memerlukan operator set kondisi.

⚠ Important

Kunci konteks multivaluasi memerlukan operator set kondisi. Jangan gunakan operator set kondisi `ForAllValues` atau `ForAnyValue` dengan kunci konteks bernilai tunggal. Untuk mempelajari selengkapnya tentang operator set kondisi, lihat [Kunci konteks multivaluasi](#).

Klasifikasi Single-valued dan Multivalued disertakan dalam deskripsi setiap kunci konteks kondisi sebagai tipe Nilai dalam topik. [AWS kunci konteks kondisi global Referensi Otorisasi Layanan](#) menggunakan klasifikasi tipe nilai yang berbeda untuk kunci konteks multivaluasi dalam format berikut: `ArrayOf` awalan diikuti oleh tipe kategori operator kondisi. Misalnya, `ArrayOfString` atau `ArrayOfARN`.

Misalnya, permintaan dapat berasal dari paling banyak satu titik akhir VPC, [the section called "SourceVpce"](#) begitu juga kunci konteks bernilai tunggal. Karena layanan dapat memiliki lebih dari satu nama utama layanan yang dimiliki oleh layanan, [aws: PrincipalServiceNamesList](#) adalah kunci konteks multivalued.

Anda dapat menggunakan kunci konteks bernilai tunggal yang tersedia sebagai variabel kebijakan. Anda tidak dapat menggunakan kunci konteks multivalued sebagai variabel kebijakan. Untuk informasi lebih lanjut tentang variabel-variabel kebijakan, lihat [Elemen kebijakan IAM: Variabel dan tag](#).

Kunci konteks multivaluasi memerlukan operator `ForAllValues` set kondisi atau `ForAnyValue` Kunci konteks yang menyertakan pasangan kunci-nilai seperti [the section called "RequestTag"](#) dan [the section called "ResourceTag"](#) dapat menyebabkan kebingungan karena bisa ada beberapa *tag-key* nilai. Tetapi karena masing-masing hanya *tag-key* dapat memiliki satu nilai, `aws:RequestTag` dan keduanya `aws:ResourceTag` merupakan kunci konteks bernilai tunggal. Menggunakan operator set kondisi dengan kunci konteks bernilai tunggal dapat menyebabkan kebijakan yang terlalu permisif.

Kunci konteks multivaluasi

Untuk membandingkan kunci konteks kondisi Anda dengan kunci [konteks permintaan](#) dengan beberapa nilai, Anda harus menggunakan `ForAllValues` atau `ForAnyValue` mengatur operator. Operator set ini digunakan untuk membandingkan dua set nilai, seperti kumpulan tag dalam permintaan dan kumpulan tag dalam kondisi kebijakan.

Qualifier `ForAllValues` dan `ForAnyValue` qualifier menambahkan fungsionalitas set-operation ke operator kondisi sehingga Anda dapat menguji kunci konteks permintaan dengan beberapa nilai terhadap beberapa nilai kunci konteks dalam kondisi kebijakan. [Selain itu, jika Anda menyertakan kunci konteks string multivalued dalam kebijakan Anda dengan wildcard atau variabel, Anda juga harus menggunakan operator kondisi. `StringLike` Beberapa nilai kunci kondisi harus diapit dalam tanda kurung seperti array.](#) Sebagai contoh, `"Key2": ["Value2A", "Value2B"]`.

- `ForAllValues`— Qualifier ini menguji apakah nilai setiap anggota dari set permintaan adalah subset dari set kunci konteks kondisi. Kondisi mengembalikan nilai `true` jika setiap nilai kunci konteks dalam permintaan cocok dengan setidaknya satu nilai kunci konteks dalam kebijakan. Ini juga mengembalikan `true` jika tidak ada kunci konteks dalam permintaan, atau jika nilai kunci konteks menyelesaikan dataset `null`, seperti string kosong. Untuk mencegah kunci konteks atau kunci konteks yang hilang dengan nilai kosong dievaluasi menjadi `true`, Anda dapat menyertakan operator `NotNull` kondisi dalam kebijakan Anda dengan nilai palsu untuk memeriksa apakah kunci konteks ada dan nilainya bukan `null`.

Important

Berhati-hatilah jika Anda menggunakan `ForAllValues` dengan `Allow` efek karena bisa terlalu permisif jika keberadaan kunci konteks yang hilang atau kunci konteks dengan nilai kosong dalam konteks permintaan tidak terduga. Anda dapat menyertakan operator `NotNull` kondisi dalam kebijakan Anda dengan nilai palsu untuk memeriksa apakah kunci konteks ada dan nilainya bukan `null`. Sebagai contoh, lihat [Mengontrol akses berdasarkan kunci tanda](#).

- `ForAnyValue`— Kualifikasi ini menguji apakah setidaknya satu anggota dari kumpulan nilai kunci konteks permintaan cocok dengan setidaknya satu anggota kumpulan nilai kunci konteks dalam kondisi kebijakan Anda. Kunci konteks mengembalikan nilai `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 dataset `null`, kondisi mengembalikan `false`.

Note

Perbedaan antara kunci konteks bernilai tunggal dan multivalued tergantung pada jumlah nilai dalam konteks permintaan, bukan jumlah nilai dalam kondisi kebijakan.

Contoh kebijakan kondisi

Dalam kebijakan IAM, Anda dapat menentukan beberapa nilai untuk kunci konteks bernilai tunggal dan multivalued untuk perbandingan terhadap 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 konteks bernilai tunggal

- Beberapa blok kondisi dengan kunci konteks bernilai tunggal. ([Lihat contoh ini.](#))
- Satu blok kondisi dengan beberapa kunci konteks bernilai tunggal dan nilai-nilai. ([Lihat contoh ini.](#))

Contoh kebijakan kondisi: Kunci konteks multivalued

- Tolak kebijakan dengan operator set kondisi `ForAllValues`. ([Lihat contoh ini.](#))
- Tolak kebijakan dengan operator set kondisi `ForAnyValue`. ([Lihat contoh ini.](#))

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 kebijakan berbasis identitas berikut menyangkal penggunaan tindakan penandaan IAM saat awalan kunci tag tertentu disertakan dalam permintaan. Setiap nilai untuk kunci konteks `aws:TagKeys` menyertakan wildcard (*) untuk pencocokan string parsial. Kebijakan ini menyertakan operator yang `ForAllValues` disetel dengan kunci konteks `aws:TagKeys` karena kunci konteks permintaan dapat menyertakan beberapa nilai. Agar kunci konteks `aws:TagKeys` mengembalikan nilai `true`, setiap nilai dalam permintaan harus cocok dengan setidaknya satu nilai dalam kebijakan.

Operator `ForAllValues` set juga mengembalikan `true` jika tidak ada kunci konteks dalam permintaan, atau jika nilai kunci konteks menyelesaikan dataset null, seperti string kosong. Untuk

mencegah kunci konteks atau kunci konteks yang hilang dengan nilai kosong dievaluasi menjadi true, sertakan operator `Null` kondisi dalam kebijakan Anda dengan nilai `false` untuk memeriksa apakah kunci konteks dalam permintaan ada dan nilainya bukan null.

Important

Kebijakan ini tidak mengizinkan tindakan apa pun. Gunakan kebijakan ini bersama dengan kebijakan lain yang mengizinkan tindakan tertentu.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyRestrictedTags",
      "Effect": "Deny",
      "Action": [
        "iam:Tag*",
        "iam:Untag*"
      ],
      "Resource": [
        "*"
      ],
      "Condition": {
        "Null": {
          "aws:TagKeys": "false"
        },
        "ForAllValues:StringLike": {
          "aws:TagKeys": [
            "key1*",
            "key2*",
            "key3*"
          ]
        }
      }
    }
  ]
}
```

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.

```
{
  "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": ["environment", "webserver"]
        }
      }
    }
  ]
}
```

Contoh kebijakan kunci konteks bernilai tunggal

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 satu kunci konteks, semua kunci konteks harus diselesaikan ke true agar Deny efek yang diinginkan Allow atau dipanggil. Saat Anda menggunakan operator kondisi pencocokan yang dinegasikan, logika evaluasi nilai kondisi dibalik.

Contoh berikut memungkinkan pengguna menciptakan volume EC2 dan menerapkan tag pada volume selama penciptaan volume. Konteks permintaan harus menyertakan nilai untuk kunci konteks `saws:RequestTag/project`, dan nilai untuk kunci konteks `aws:ResourceTag/environment` dapat berupa apa saja kecuali produksi.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "ec2:CreateVolume",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "ec2:CreateTags",
      "Resource": "arn:aws:ec2:::volume/*",
      "Condition": {
        "StringLike": {
          "aws:RequestTag/project": "*"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": "ec2:CreateTags",
      "Resource": "arn:aws:ec2:region:account:*/*",
      "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 sumber daya.

```
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 dan nilai konteks bernilai tunggal

Ketika blok kondisi berisi beberapa kunci konteks dan setiap kunci konteks memiliki beberapa nilai, setiap kunci konteks harus menyelesaikan ke true untuk setidaknya satu nilai kunci untuk yang diinginkan Allow atau Deny efek yang akan 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 Amazon Elastic Container Service cluster.

- Konteks permintaan harus menyertakan production OR pre-prod untuk kunci aws:RequestTag/environment konteks DAN.
- Kunci ecs:cluster konteks memastikan bahwa tugas dijalankan pada kluster default1 OR default2 ARN ECS.

```
{  
  "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"
      ]
    }
  }
}
```

Elemen kebijakan IAM: Variabel dan tag

Gunakan variabel kebijakan AWS Identity and Access Management (IAM) sebagai placeholder ketika Anda tidak mengetahui nilai pasti dari sumber daya atau kunci kondisi saat Anda 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 operator kondisi.](#)

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 DOC-EXAMPLE-BUCKET untuk marketing proyek.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["s3:ListBucket"],
      "Resource": ["arn:aws:s3:::DOC-EXAMPLE-BUCKET"],
      "Condition": {"StringLike": {"s3:prefix": ["marketing/*"]}}
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": ["arn:aws:s3:::DOC-EXAMPLE-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, kami sarankan Anda membuat 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 kurawal kurawal (`{ }`) yang menyertakan nama variabel nilai dari permintaan.

Ketika kebijakan dievaluasi, variabel kebijakan diganti dengan nilai yang berasal dari kunci konteks bersyarat yang diteruskan dalam permintaan. [Variabel dapat digunakan dalam kebijakan berbasis identitas, kebijakan sumber daya, kebijakan kontrol layanan, kebijakan sesi, dan kebijakan titik akhir VPC](#). Kebijakan berbasis identitas 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 Otorisasi Layanan](#). Dalam keadaan tertentu, Anda tidak dapat mengisi kunci konteks kondisi global dengan nilai. Untuk mempelajari selengkapnya 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 apa pun sebagai variabel. Anda tidak dapat menggunakan kunci kondisi multivalued sebagai variabel.

Contoh berikut menunjukkan kebijakan untuk peran IAM atau pengguna yang menggantikan nama sumber daya tertentu dengan variabel kebijakan. Anda dapat menggunakan kembali kebijakan ini dengan memanfaatkan kunci `aws:PrincipalTag` kondisi. Jika kebijakan ini dievaluasi, hanya `${aws:PrincipalTag/team}` mengizinkan tindakan jika nama bucket diakhiri dengan nama tim dari tag team utama.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["s3:ListBucket"],
      "Resource": ["arn:aws:s3:::DOC-EXAMPLE-BUCKET"],
      "Condition": {"StringLike": {"s3:prefix": ["${aws:PrincipalTag/team}/*"]}}
```

```
  },
  {
    "Effect": "Allow",
    "Action": [
      "s3:GetObject",
      "s3:PutObject"
    ],
    "Resource": ["arn:aws:s3:::DOC-EXAMPLE-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 [aws:PrincipalTag/tag-kunci](#) di 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 prinsipal untuk mendapatkan objek dari jalur /David dari bucket S3 terlihat seperti ini:

```
{
```

```
"Version": "2012-10-17",
"Statement": [{
  "Effect": "Allow",
  "Action": ["s3:GetObject"],
  "Resource": ["arn:aws:s3::DOC-EXAMPLE-BUCKET/David/*"]
}]
}
```

Jika kebijakan ini dilampirkan ke pengguna `David`, 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 `aws:PrincipalTag` cocok dengan nilai kunci tag yang diteruskan dalam permintaan `owner`.

```
{
  "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 pada peran IAM yang diasumsikan oleh Manajer Produk menggunakan kredensi keamanan sementara. Saat pengguna membuat permintaan untuk menambahkan objek Amazon S3, IAM mengganti nilai dept tag dari permintaan `${aws:PrincipalTag}` variabel saat ini dan mengevaluasi kebijakan.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "AllowOnlyDeptS3Prefix",
    "Effect": "Allow",
    "Action": ["s3:GetObject"],
    "Resource": ["arn:aws:s3:::DOC-EXAMPLE-BUCKET/${aws:PrincipalTag/dept}/*"],
  }
]
```

Tanda sebagai variabel kebijakan

Di beberapa AWS layanan, Anda dapat melampirkan atribut kustom Anda sendiri ke sumber daya yang dibuat oleh layanan tersebut. Misalnya, Anda dapat menerapkan tag ke bucket Amazon S3 atau ke 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 [Menandai sumber daya IAM](#). 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 AWS Management Console .

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 dari 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:::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 memodifikasi sumber daya mereka. Selain itu, menggunakan [kecocokan wildcard *](#), mereka dapat mengizinkan sufiks nama sumber daya khusus.

```
{
  "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 IAM JSON: 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.

```
{
  "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 serupa. 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 konteks kondisi global masuk [AWS kunci konteks kondisi global](#).

Bila Anda menggunakan variabel tanpa nilai dalam elemen kondisi kebijakan IAM, [Elemen kebijakan IAM JSON: Operator kondisi](#) `StringEquals` atau `StringLike` tidak cocok, dan pernyataan kebijakan tidak berlaku.

Operator kondisi terbalik `StringNotLike` menyukai `StringNotEquals` atau mencocokkan dengan nilai nol, karena nilai kunci kondisi yang mereka uji tidak sama dengan atau seperti nilai nol efektif.

Dalam contoh berikut, `aws:principaltag/Team` harus sama dengan `s3:ExistingObjectTag/Team` mengizinkan akses. Akses ditolak secara eksplisit ketika tidak `aws:principaltag/Team` disetel. Jika variabel yang tidak memiliki nilai dalam konteks otorisasi digunakan sebagai bagian dari `Resource` atau `NotResource` elemen kebijakan, sumber daya yang menyertakan variabel kebijakan tanpa nilai tidak akan cocok dengan sumber daya apa pun.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::/example-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. Bila 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:user-id`, 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:user-id:` Akun AWS ID
- `aws:PrincipalType:` Account
- Pengguna IAM
 - `aws:username:` *IAM-user-name*
 - `aws:user-id:` ID unik
 - `aws:PrincipalType:` User
- Pengguna federasi
 - `aws:username:` (tidak ada)
 - `aws:user-id:` *akun: caller-specified-name*
 - `aws:PrincipalType:` FederatedUser
- Pengguna federasi web dan pengguna federasi SAMP

 Note

Untuk informasi tentang kunci kebijakan yang tersedia saat Anda menggunakan federasi OIDC, lihat [Identifikasi pengguna dengan federasi OIDC](#)

- `aws:username:` (tidak ada)
- `aws:user-id:` (tidak ada)
- `aws:PrincipalType:` AssumedRole
- Peran yang diasumsikan
 - `aws:username:` (tidak ada)
 - `aws:user-id:` *role-id: caller-specified-role-name*
 - `aws:PrincipalType:` Assumed role
- Peran yang ditetapkan ke instans Amazon EC2
 - `aws:username:` (tidak ada)
 - `aws:user-id:` *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 mencocokkannya gagal dan menyebabkan pernyataan tersebut menjadi tidak valid.
- *role-id* adalah pengidentifikasi unik yang ditetapkan untuk setiap peran saat dibuat. Anda dapat menampilkan ID peran dengan AWS CLI perintah: `aws iam get-role --role-name rolename`
- *caller-specified-name* dan *caller-specified-role-name* merupakan nama yang diteruskan oleh proses panggilan (seperti aplikasi atau layanan) ketika melakukan panggilan untuk mendapatkan kredensi sementara.
- *ec2-instance-id* adalah nilai yang ditetapkan ke 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 yang tersedia dalam permintaan bagi pengguna gabungan

Pengguna gabungan 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 SAMP pada dasarnya dapat bertukar informasi identitas AWS untuk kredensial keamanan sementara. Kredensi 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 mengautentikasi pengguna menggunakan penyedia identitas terkenal seperti Login with Amazon, Amazon Cognito, Facebook, atau Google. Kemudian aplikasi dapat menggunakan informasi

otentikasi 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 AWS SDK seluler. Untuk informasi selengkapnya, lihat hal berikut:

- [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 prinsipal diberi tag `team=yellow`, mereka dapat mengakses bucket `ExampleCorp's Amazon S3 bernama DOC-EXAMPLE-BUCKET-yellow`. Kebijakan dengan sumber daya ini memungkinkan anggota tim untuk mengakses keranjang tim mereka, tetapi tidak dengan tim lain. Untuk pengguna tanpa tag tim, ini menetapkan nilai default `company-wide` untuk nama bucket. Pengguna ini hanya dapat mengakses `DOC-EXAMPLE-BUCKET-company-wide` bucket di mana mereka dapat melihat informasi yang luas, seperti instruksi untuk bergabung dengan tim.

```
"Resource": "arn:aws:s3::DOC-EXAMPLE-BUCKET-${aws:PrincipalTag/team, 'company-wide'}"
```

Untuk informasi selengkapnya

Untuk informasi selengkapnya tentang kebijakan, lihat hal berikut:

- [Kebijakan dan Izin di IAM](#)
- [Contoh kebijakan berbasis identitas IAM](#)
- [Referensi elemen kebijakan IAM JSON](#)
- [Logika evaluasi kebijakan](#)
- [Federasi OIDC](#)

Elemen kebijakan IAM JSON: Tipe 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 dalam UTF-8. Untuk informasi tentang tipe data JSON, kunjungi [RFC 4627](#).

Tipe	JSON
String	String
Bulat	Bilangan

Tipe	JSON
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 prinsipal mencoba menggunakan AWS Management Console, API AWS, atau AWS CLI, dengan permintaan yang dikirim prinsipal ke AWS. Saat layanan AWS menerima permintaan, AWS menyelesaikan 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.
3. [Mengevaluasi kebijakan dalam satu akun](#)— AWS mengevaluasi semua jenis kebijakan, yang mempengaruhi urutan kebijakan dievaluasi.
4. [Menentukan apakah permintaan diizinkan atau ditolak dalam sebuah akun.](#)— AWS kemudian memproses kebijakan terhadap konteks permintaan untuk menentukan apakah permintaan diizinkan atau ditolak.

Memproses konteks permintaan

AWS memproses permintaan untuk mengumpulkan informasi berikut menjadi konteks permintaan:

- Tindakan (atau operasi) – Tindakan atau operasi yang ingin dilakukan oleh prinsipal.
- Sumber daya - Objek AWS sumber daya tempat tindakan atau operasi dilakukan.
- Prinsipal – Pengguna, peran, pengguna gabungan, atau aplikasi yang mengirimkan permintaan tersebut. Informasi tentang prinsipal mencakup kebijakan yang terkait dengan prinsipal tersebut
- Data lingkungan – Informasi tentang alamat IP, agen pengguna, status yang diaktifkan SSL, atau waktu hari.
- Data sumber daya – Data terkait sumber daya yang diminta. Hal ini dapat mencakup informasi seperti nama tabel DynamoDB atau tag pada instans Amazon EC2.

Kemudian, AWS menggunakan informasi ini untuk menemukan kebijakan yang berlaku pada konteks permintaan.

Mengevaluasi kebijakan dalam satu akun

Cara AWS mengevaluasi kebijakan bergantung pada tipe kebijakan yang berlaku pada konteks permintaan. Jenis kebijakan berikut, yang tercantum dalam urutan frekuensi, tersedia untuk digunakan dalam satu Akun AWS. Untuk informasi selengkapnya tentang tipe kebijakan ini, lihat [Kebijakan dan Izin di IAM](#). Untuk mempelajari cara AWS mengevaluasi kebijakan untuk akses lintas akun, lihat [Logika evaluasi kebijakan lintas akun](#).

1. 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, AWS memeriksa semua kebijakan tersebut untuk setidaknya satu Allow.
2. Kebijakan berbasis sumber daya - Kebijakan berbasis sumber daya memberikan izin kepada prinsipal (prinsipal akun, pengguna, peran, dan sesi seperti sesi peran dan pengguna federasi IAM) yang ditetapkan sebagai prinsipal. Izin menentukan apa yang dapat dilakukan oleh prinsipal dengan sumber daya yang memiliki kebijakan tersebut. Jika kebijakan berbasis sumber daya dan kebijakan berbasis identitas berlaku untuk permintaan, AWS memeriksa semua kebijakan untuk setidaknya satu Allow. Ketika kebijakan berbasis sumber daya dievaluasi, ARN utama yang ditentukan dalam kebijakan menentukan apakah penolakan implisit dalam jenis kebijakan lain berlaku untuk keputusan akhir.
3. Batas izin IAM – Batas izin adalah fitur lanjutan yang mengatur izin maksimum yang dapat diberikan oleh kebijakan berbasis identitas kepada entitas IAM (pengguna atau peran). Saat Anda menetapkan batas izin untuk suatu entitas, entitas hanya dapat melakukan tindakan yang diizinkan oleh kedua kebijakan berbasis identitas dan batas izinnya. Dalam beberapa kasus, penolakan

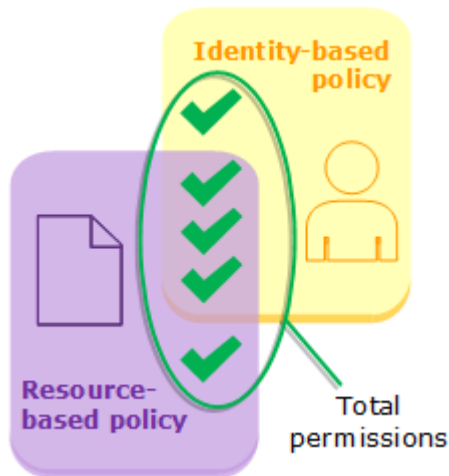
implisit dalam batas izin dapat membatasi izin yang diberikan oleh kebijakan berbasis sumber daya. Untuk mempelajari lebih lanjut, lihat [Menentukan apakah permintaan diizinkan atau ditolak dalam sebuah akun](#). nanti dalam topik ini.

4. AWS Organizations kebijakan kontrol layanan (SCP) - Organisasi SCP menentukan izin maksimum untuk organisasi atau unit organisasi (OU). Maksimum SCP berlaku untuk prinsipal dalam akun anggota, termasuk setiap Pengguna root akun AWS. Jika SCP ada, kebijakan berbasis identitas dan berbasis sumber daya memberikan izin kepada prinsipal dalam akun anggota hanya jika kebijakan tersebut dan SCP mengizinkan tindakan. Jika kedua batas izin dan SCP ada, maka batas, SCP, dan kebijakan berbasis identitas semuanya harus mengizinkan tindakan.
5. Kebijakan sesi – Kebijakan sesi adalah kebijakan tingkat lanjut yang Anda teruskan sebagai parameter saat Anda secara terprogram membuat sesi sementara untuk pengguna peran atau 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` Kebijakan berbasis sumber daya memiliki efek yang berbeda pada evaluasi izin kebijakan sesi. Perbedaannya bergantung pada apakah ARN pengguna atau peran atau ARN sesi dicantumkan sebagai prinsip utama dalam kebijakan berbasis sumber daya. Untuk informasi selengkapnya, lihat [Kebijakan sesi](#).

Ingat, penolakan secara tegas dalam salah satu kebijakan ini membatalkan izin.

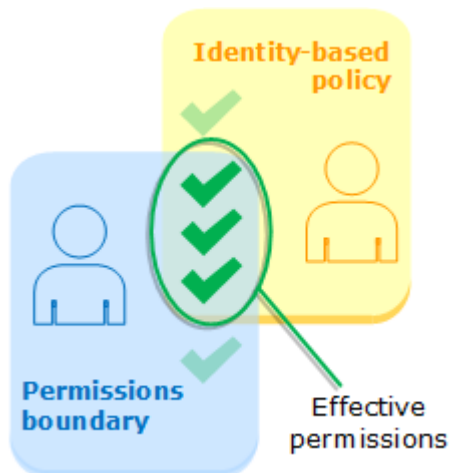
Mengevaluasi kebijakan berbasis identitas dengan kebijakan berbasis sumber daya

Kebijakan berbasis identitas dan kebijakan berbasis sumber daya memberikan izin kepada identitas atau sumber daya yang melekat padanya. 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 berbasis sumber daya. Izin yang dihasilkan adalah izin total dari dua tipe. Jika tindakan diizinkan oleh kebijakan berbasis identitas, kebijakan berbasis sumber daya, atau keduanya, AWS mengizinkan tindakan tersebut. 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 bagi pengguna, izin yang dihasilkan adalah perpotongan dari dua kategori tersebut. 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 Organisasi SCP

Ketika pengguna memiliki akun yang merupakan anggota dari sebuah organisasi, izin yang dihasilkan adalah perpotongan kebijakan pengguna dan SCP. Ini berarti bahwa tindakan harus diizinkan oleh kebijakan berbasis identitas dan SCP. Penolakan secara tegas dalam salah satu kebijakan ini membatalkan izin.



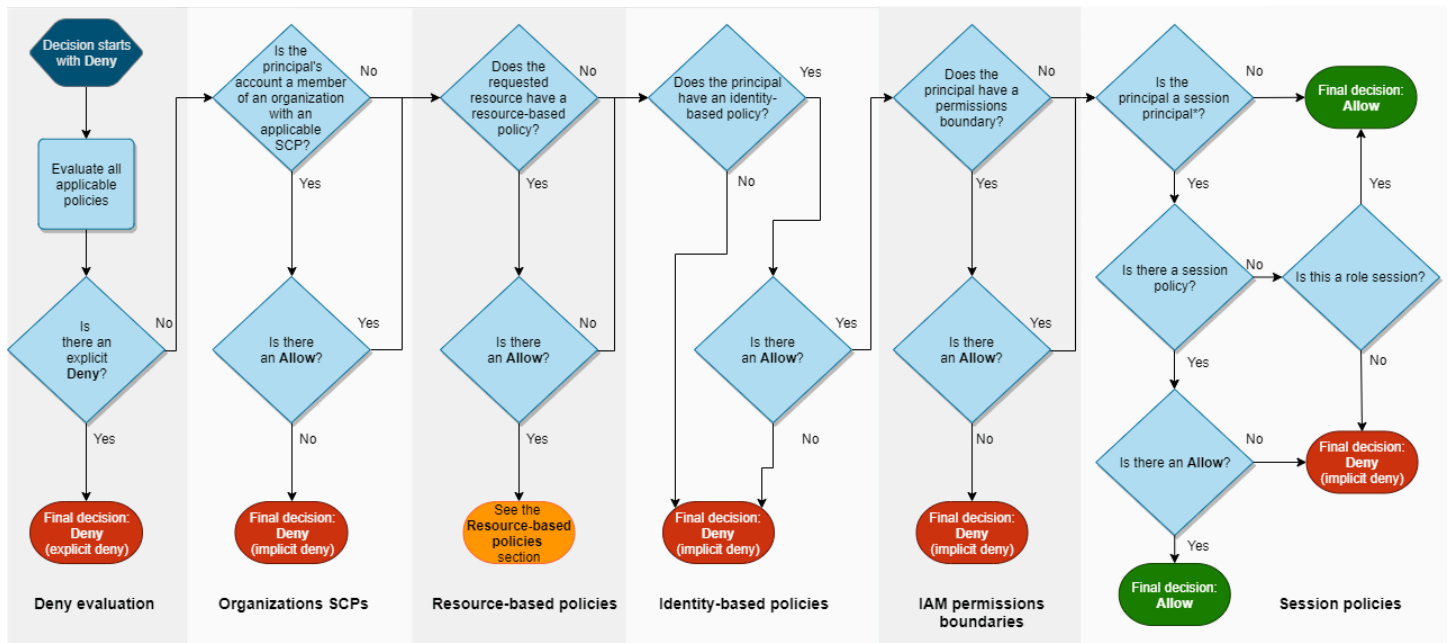
Anda dapat mempelajari [apakah akun Anda adalah anggota organisasi](#) di AWS Organizations. Anggota organisasi dapat dipengaruhi oleh SCP. Untuk melihat data ini menggunakan AWS CLI perintah atau operasi AWS API, Anda harus memiliki izin untuk `organizations:DescribeOrganization` tindakan untuk entitas Organisasi Anda. Anda harus memiliki izin tambahan untuk melakukan operasi di konsol Organisasi. Untuk mempelajari apakah SCP menolak akses ke permintaan tertentu, atau untuk mengubah izin efektif Anda, hubungi administrator AWS Organizations Anda.

Menentukan apakah permintaan diizinkan atau ditolak dalam sebuah akun.

Anggap prinsipal mengirim permintaan ke AWS untuk mengakses sumber daya dalam akun yang sama dengan entitas prinsipal. Kode AWS penegakan memutuskan apakah permintaan harus diizinkan atau ditolak. AWS mengevaluasi semua kebijakan yang berlaku untuk konteks permintaan. Berikut ini adalah ringkasan logika AWS evaluasi untuk kebijakan dalam satu akun.

- Secara default, semua permintaan secara implisit ditolak dengan pengecualian Pengguna root akun AWS, yang memiliki akses penuh.
- Izin eksplisit dalam kebijakan berbasis identitas atau berbasis sumber daya akan membatalkan pengaturan default ini.
- Jika ada batasan izin, Organisasi SCP, atau kebijakan sesi ada, hal ini dapat mengesampingkan izin dengan penolakan implisit.
- Penolakan secara tegas dalam kebijakan apa pun akan mengesampingkan izin apa pun.

Bagan alir berikut ini memberikan perincian tentang cara pengambilan keputusan. Diagram alir ini tidak mencakup dampak kebijakan berbasis sumber daya dan penolakan implisit dalam jenis kebijakan lainnya.



1. Evaluasi penolakan – Secara default, semua permintaan ditolak. Ini disebut [penolakan implisit](#). Kode penerapan AWS mengevaluasi semua kebijakan di dalam akun yang berlaku untuk permintaan. Ini termasuk AWS Organizations SCP, 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 mengembalikan keputusan akhir Deny. Jika tidak ada penolakan eksplisit, evaluasi kode penegakan terus berlanjut.
2. Organisasi SCP - Kemudian 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, permintaan tersebut ditolak secara eksplisit, bahkan jika penolakan tersebut implisit. Kode penegakan mengembalikan keputusan akhir Deny. Jika tidak ada SCP, atau jika SCP mengizinkan tindakan yang diminta, evaluasi kode penegakan berlanjut.
3. Kebijakan berbasis sumber daya - Dalam akun yang sama, kebijakan berbasis sumber daya memengaruhi evaluasi kebijakan secara berbeda tergantung pada jenis pokok yang mengakses sumber daya, dan pokok yang diizinkan dalam kebijakan berbasis sumber daya. Tergantung pada jenis pokok, Allow dalam kebijakan berbasis sumber daya dapat menghasilkan keputusan akhir Allow, bahkan jika penolakan implisit dalam kebijakan berbasis identitas, batas izin, atau kebijakan sesi hadir.

Untuk sebagian besar sumber daya, Anda hanya memerlukan izin eksplisit untuk prinsipal baik dalam kebijakan berbasis identitas atau kebijakan berbasis sumber daya untuk memberikan akses. [Kebijakan kepercayaan peran IAM dan kebijakan utama KMS adalah pengecualian untuk logika ini, karena mereka harus secara eksplisit mengizinkan akses untuk prinsipal.](#)

Logika kebijakan berbasis sumber daya berbeda dari jenis kebijakan lain jika prinsipal yang ditentukan adalah pengguna IAM, peran IAM, atau prinsipal sesi. Prinsip sesi mencakup sesi [peran IAM atau sesi pengguna gabungan IAM](#). Jika kebijakan berbasis sumber daya memberikan izin langsung kepada pengguna IAM atau prinsipal sesi yang membuat permintaan, maka penolakan implisit dalam kebijakan berbasis identitas, batas izin, atau kebijakan sesi tidak memengaruhi keputusan akhir.

Tabel berikut membantu Anda memahami dampak kebijakan berbasis sumber daya untuk berbagai jenis pokok ketika penolakan implisit hadir dalam kebijakan berbasis identitas, batas izin, dan kebijakan sesi.

Kebijakan berbasis sumber daya dan penolakan implisit dalam jenis kebijakan lain (akun yang sama)

Kepala sekolah membuat permintaan	Kebijakan berbasis sumber daya	Kebijakan berbasis identitas	Batas izin	Kebijakan Sesi	Hasil	Alasan
IAM role	Tidak berlaku	Tidak berlaku	Tidak berlaku	Tidak berlaku	Tidak berlaku	Peran itu sendiri tidak dapat membuat permintaan. Permintaan dibuat dengan sesi peran setelah peran diasumsikan.

Kepala sekolah membuat permintaan	Kebijakan berbasis sumber daya	Kebijakan berbasis identitas	Batas izin	Kebijakan Sesi	Hasil	Alasan
Sesi peran IAM	Memungkinkan peran ARN	Implisit menyangkal	Implisit menyangkal	Implisit menyangkal	MENOLAK	Batas izin dan kebijakan sesi dievaluasi sebagai bagian dari keputusan akhir. Penyangkalan implisit dalam salah satu kebijakan menghasilkan keputusan DENY.
Sesi peran IAM	Memungkinkan sesi peran ARN	Implisit menyangkal	Implisit menyangkal	Implisit menyangkal	IZINKAN	Izin diberikan langsung ke sesi. Jenis kebijakan lainnya tidak mempengaruhi keputusan.

Kepala sekolah membuat permintaan	Kebijakan berbasis sumber daya	Kebijakan berbasis identitas	Batas izin	Kebijakan Sesi	Hasil	Alasan
Pengguna IAM	Memungkinkan ARN pengguna IAM	Implisit menyangkal	Implisit menyangkal	Tidak berlaku	IZINKAN	Izin diberikan langsung kepada pengguna. Jenis kebijakan lainnya tidak mempengaruhi keputusan.
Pengguna federasi IAM () GetFederationToken	Memungkinkan ARN pengguna IAM	Implisit menyangkal	Implisit menyangkal	Implisit menyangkal	MENOLAK	Penyangkalan implisit baik dalam batas izin atau kebijakan sesi menghasilkan DENY.

Kepala sekolah membuat permintaan	Kebijakan berbasis sumber daya	Kebijakan berbasis identitas	Batas izin	Kebijakan Sesi	Hasil	Alasan
Pengguna federasi IAM () GetFederationToken	Memungkinkan IAM federasi sesi pengguna ARN	Implisit menyangkal	Implisit menyangkal	Implisit menyangkal	IZINKAN	Izin diberikan langsung ke sesi. Jenis kebijakan lainnya tidak mempengaruhi keputusan.

Kepala sekolah membuat permintaan	Kebijakan berbasis sumber daya	Kebijakan berbasis identitas	Batas izin	Kebijakan Sesi	Hasil	Alasan
pengguna root	Memungkinkan ARN pengguna root	Tidak berlaku	Tidak berlaku	Tidak berlaku	IZINKAN	Pengguna root memiliki akses lengkap dan tidak terbatas ke semua sumber daya di Anda. Akun AWS Untuk mempelajari cara mengontrol akses ke pengguna root untuk akun AWS Organizations, lihat Kebijakan kontrol layanan (SCP) di Panduan Pengguna Organisasi.

Kepala sekolah membuat permintaan	Kebijakan berbasis sumber daya	Kebijakan berbasis identitas	Batas izin	Kebijakan Sesi	Hasil	Alasan
Layanan utama AWS	Memungkinkan prinsipal AWS layanan	Tidak berlaku	Tidak berlaku	Tidak berlaku	IZINKAN	Ketika kebijakan berbasis sumber daya memberikan izin langsung kepada prinsipal AWS layanan , jenis kebijakan lainnya tidak memengaruhi keputusan tersebut.

- Peran IAM - Kebijakan berbasis sumber daya yang memberikan izin untuk peran IAM ARN dibatasi oleh penolakan implisit dalam batas izin atau kebijakan sesi.

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 peran itu sendiri.

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 ARN pengguna IAM (yang bukan sesi pengguna federasi) tidak dibatasi oleh penolakan implisit dalam kebijakan berbasis identitas atau batas izin.

Contoh IAM pengguna ARN

```
arn:aws:iam::111122223333:user/exampleuser
```

- Sesi pengguna federasi IAM - Sesi pengguna federasi IAM adalah sesi yang dibuat dengan menelepon. [GetFederationToken](#) Ketika pengguna federasi membuat permintaan, prinsipal yang membuat permintaan adalah ARN pengguna federasi dan bukan ARN dari pengguna IAM yang federasi. Dalam akun yang sama, kebijakan berbasis sumber daya yang memberikan izin kepada ARN pengguna federasi 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 berfederasi, maka permintaan yang dibuat oleh pengguna federasi selama sesi dibatasi oleh penolakan implisit dalam batas izin atau kebijakan sesi.

Contoh IAM federasi sesi pengguna ARN

```
arn:aws:sts::111122223333:federated-user/exampleuser
```

4. Kebijakan berbasis identitas – Kemudian kode tersebut 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 memungkinkan tindakan yang diminta, maka permintaan tersebut secara implisit ditolak dan kode mengembalikan keputusan akhir Deny. Jika ada pernyataan dalam kebijakan berbasis identitas yang berlaku mengizinkan tindakan yang diminta, kode akan berlanjut.

5. Batas izin IAM - Kode kemudian memeriksa apakah entitas IAM yang digunakan oleh prinsipal memiliki batas izin. Jika kebijakan yang digunakan untuk menetapkan batas izin tidak mengizinkan tindakan yang diminta, maka permintaan tersebut ditolak secara implisit. Kode tersebut memberikan keputusan akhir Tolak. Jika tidak ada batasan izin, atau jika batas izin memungkinkan tindakan yang diminta, kode berlanjut.
6. Kebijakan sesi - Kode kemudian memeriksa apakah prinsipal adalah kepala sekolah sesi. Prinsip sesi mencakup sesi peran IAM atau sesi pengguna gabungan IAM. Jika kepala sekolah bukan kepala sekolah sesi, kode penegakan mengembalikan keputusan akhir Izinkan.

Untuk prinsipal sesi, kode memeriksa apakah kebijakan sesi diteruskan dalam permintaan. Anda dapat meneruskan kebijakan sesi saat menggunakan AWS CLI atau AWS API untuk mendapatkan kredensi sementara untuk peran atau pengguna gabungan IAM.

- Jika kebijakan sesi hadir dan tidak mengizinkan tindakan yang diminta, maka permintaan tersebut secara implisit ditolak. Kode tersebut memberikan keputusan akhir Tolak.
 - Jika tidak ada kebijakan sesi, kode akan memeriksa apakah prinsipal adalah sesi peran. Jika prinsipal adalah sesi peran, maka permintaan tersebut Diizinkan. Jika tidak, permintaan secara implisit ditolak dan kode mengembalikan keputusan akhir dari Deny.
 - Jika kebijakan sesi hadir dan memungkinkan tindakan yang diminta, maka kode penegakan mengembalikan keputusan akhir Izinkan.
7. Kesalahan - Jika kode AWS penegakan menemukan kesalahan pada titik mana pun selama evaluasi, maka itu menghasilkan pengecualian dan menutup.

Contoh evaluasi kebijakan berbasis identitas dan kebijakan berbasis sumber daya

Tipe kebijakan yang paling umum adalah kebijakan berbasis identitas dan kebijakan berbasis sumber daya. Saat akses ke sumber daya diminta, AWS evaluasi semua izin yang diberikan oleh kebijakan untuk setidaknya satu Izinkan dalam akun yang sama. Penafian eksplisit dalam salah satu kebijakan menimpa allow.

Important

Jika kebijakan berbasis identitas atau kebijakan berbasis sumber daya dalam akun yang sama mengizinkan permintaan dan yang lainnya tidak, permintaan tersebut tetap diizinkan.

Asumsikan bahwa Carlos memiliki nama pengguna `carlossalazar` dan dia mencoba menyimpan file ke bucket Amazon S3 `carlossalazar-logs`.

Juga asumsikan bahwa kebijakan berikut ini diberlakukan pada pengguna IAM `carlossalazar`.

```
{
  "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:::carlossalazar/*",
        "arn:aws:s3:::carlossalazar"
      ]
    },
    {
      "Sid": "DenyS3Logs",
      "Effect": "Deny",
      "Action": "s3:*",
      "Resource": "arn:aws:s3::*:log*"
    }
  ]
}
```

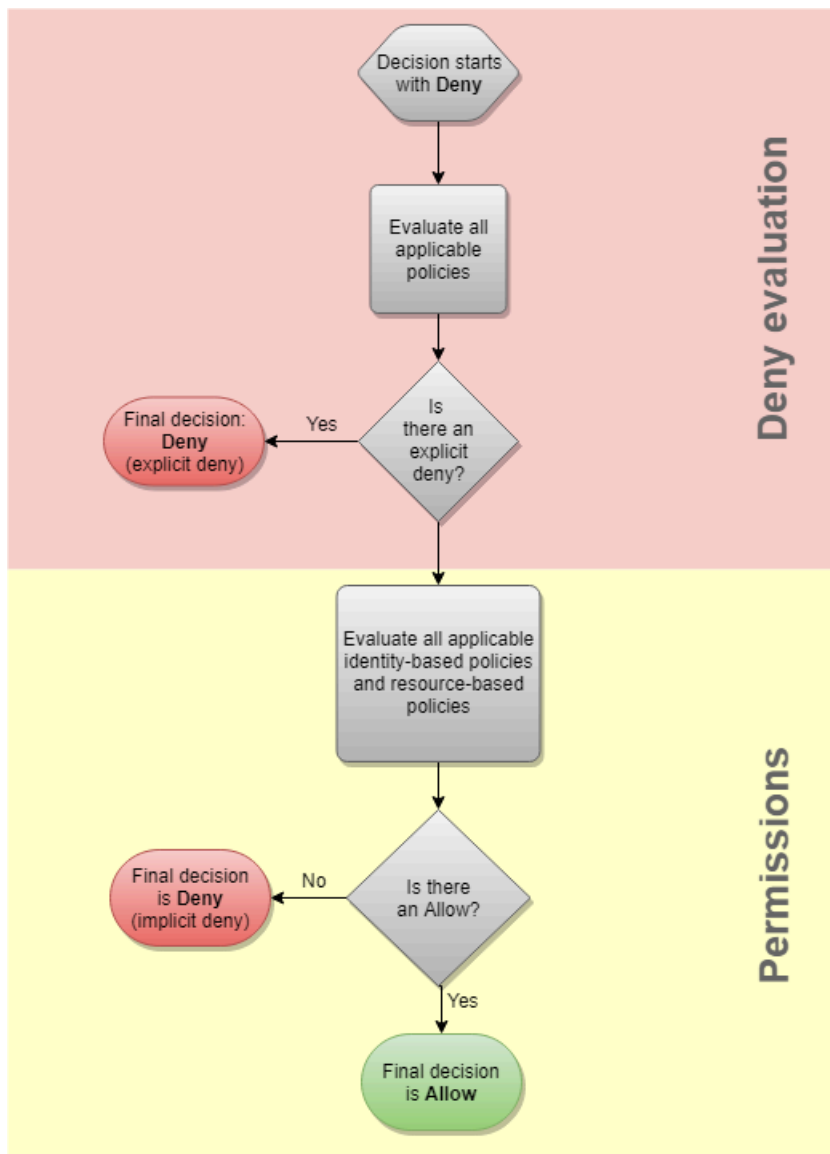
Pernyataan `AllowS3ListRead` dalam kebijakan ini memungkinkan Carlos melihat daftar semua bucket di akun. Pernyataan `AllowS3Self` memungkinkan Carlos mendapatkan akses penuh ke bucket dengan nama yang sama dengan nama penggunanya. Pernyataan `DenyS3Logs` menolak akses Carlos ke setiap bucket S3 mana pun dengan `log` dalam namanya.

Selain itu, kebijakan berbasis sumber daya berikut (disebut kebijakan buket) dilampirkan ke bucket `carlossalazar`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:user/carlossalazar"
      },
      "Action": "s3:*",
      "Resource": [
        "arn:aws:s3:::carlossalazar/*",
        "arn:aws:s3:::carlossalazar"
      ]
    }
  ]
}
```

Kebijakan ini menetapkan bahwa hanya pengguna `carlossalazar` yang dapat mengakses bucket `carlossalazar`.

Saat Carlos meminta untuk menyimpan file ke bucket `carlossalazar-logs`, AWS menentukan kebijakan apa yang berlaku terhadap 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.



Pertama-tama, AWS memeriksa pernyataan Deny yang berlaku untuk konteks permintaan. Ini menemukan satu, karena kebijakan berbasis identitas secara eksplisit menolak akses Carlos ke bucket S3 yang digunakan untuk logging. Akses Carlos ditolak.

Asumsikan dia kemudian menyadari kesalahannya dan mencoba menyimpan file tersebut ke bucket `carlossalazar`. AWS memeriksa pernyataan Deny dan tidak menemukan apa pun. Kemudian memeriksa kebijakan izin. Baik kebijakan berbasis identitas maupun kebijakan berbasis sumber daya mengizinkan permintaan tersebut. Oleh karena itu, AWS mengizinkan permintaan tersebut. 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.

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 akses secara default, mereka harus secara eksplisit diizinkan untuk melakukan tindakan. Jika tidak, akses 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. AllowGetListPernyataan ini memungkinkan akses hanya-baca ke tindakan IAM yang dimulai dengan awalan dan. Get List Semua tindakan lain dalam IAM, seperti secara implisit **iam:CreatePolicy** ditolak. DenyReportsPernyataan 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, misalnya, permintaan terkait laporan masih ditolak karena iam:GenerateCredentialReport penolakan eksplisit ini.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowGetList",
      "Effect": "Allow",
      "Action": [
        "iam:Get*",
        "iam:List*"
      ],
      "Resource": "*"
    },
    {
      "Sid": "DenyReports",
      "Effect": "Deny",
      "Action": "iam:*Report",
      "Resource": "*"
    }
  ]
}
```

```
    }  
  ]  
}
```

Logika evaluasi kebijakan lintas akun

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 ke 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, pengguna gabungan, peran IAM, atau sesi paralel yang diterimanya. Anda juga dapat menentukan layanan AWS sebagai prinsipal. Untuk informasi selengkapnya, lihat [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 atau dengan memungkinkan akses ke semua 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](#).

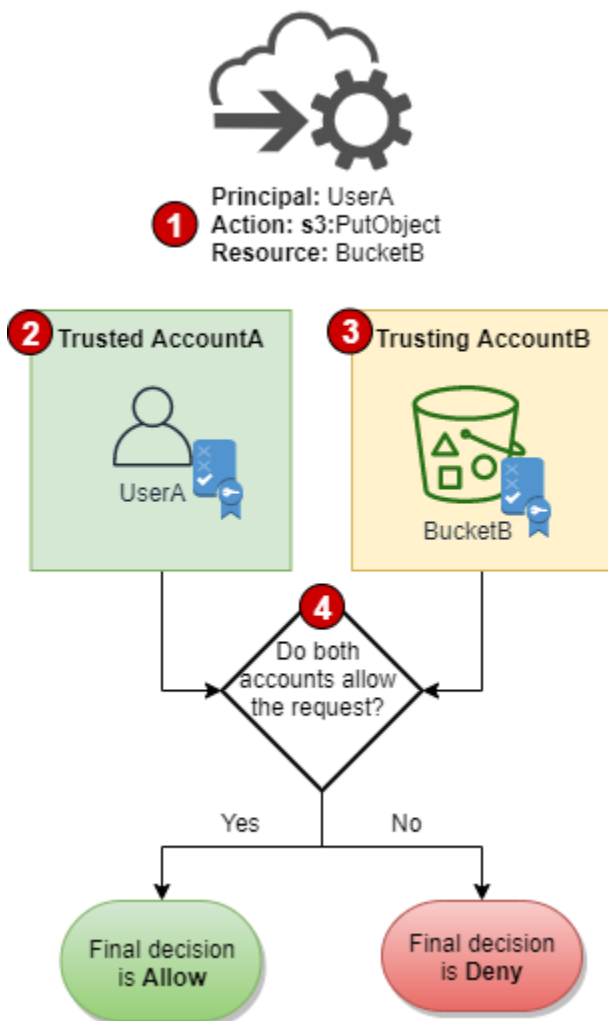
Important

Layanan lain dapat memengaruhi logika evaluasi kebijakan. Misalnya, AWS Organizations mendukung [kebijakan kontrol layanan](#) yang dapat diterapkan ke satu atau beberapa akun prinsipal. AWS Resource Access Manager mendukung [fragmen kebijakan](#) yang mengontrol tindakan mana yang diizinkan untuk dilakukan oleh prinsipal pada sumber daya yang dibagikan dengan mereka.

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 melakukan permintaan lintas akun, AWS melakukan dua evaluasi. AWS mengevaluasi permintaan dalam akun kepercayaan dan akun tepercaya. Untuk informasi selengkapnya tentang bagaimana permintaan dievaluasi di satu akun, lihat [Menentukan apakah permintaan diizinkan atau ditolak dalam sebuah akun](#). 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 dalam satu akun](#).
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 dalam satu akun](#).
4. AWS hanya mengizinkan permintaan jika kedua evaluasi kebijakan akun mengizinkan permintaan tersebut.

Contoh evaluasi kebijakan lintas akun

Contoh berikut menunjukkan skenario di mana pengguna dalam satu akun diberikan izin oleh kebijakan berbasis sumber daya di akun kedua.

Asumsikan bahwa Carlos adalah seorang pengembang dengan pengguna IAM bernama `carloossalazar` di akun `111111111111`. Dia ingin menyimpan file ke bucket Amazon S3 `Production-logs` di akun `222222222222`.

Juga asumsikan bahwa kebijakan berikut ini diberlakukan pada pengguna IAM `carloossalazar`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowS3ListRead",
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    },
    {
      "Sid": "AllowS3ProductionObjectActions",
      "Effect": "Allow",
      "Action": "s3:*Object*",
      "Resource": "arn:aws:s3:::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 `Production`. Pernyataan `DenyS3Logs` menolak akses Carlos ke setiap bucket S3 mana pun dengan log dalam namanya. Pernyataan ini juga menghalangi akses ke semua objek di bucket tersebut.

Selain itu, kebijakan berbasis sumber daya berikut (disebut kebijakan bucket) diberlakukan untuk bucket `Production` di akun `222222222222`.

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject*",
        "s3:PutObject*",
        "s3:ReplicateObject",
        "s3:RestoreObject"
      ],
      "Principal": { "AWS": "arn:aws:iam::111111111111:user/carlossalazar" },
      "Resource": "arn:aws:s3::*Production/*"
    }
  ]
}

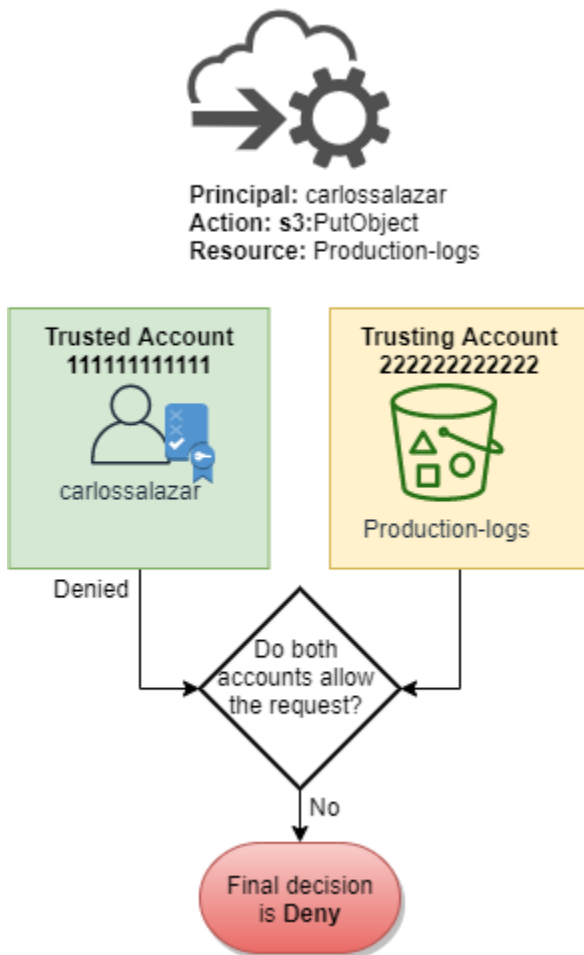
```

Kebijakan ini memungkinkan pengguna `carlossalazar` untuk mengakses objek di bucket `Production`. Dia dapat membuat dan mengedit, tetapi tidak menghapus objek dalam bucket. Dia tidak dapat mengelola bucket sendiri.

Saat Carlos meminta untuk menyimpan file ke bucket `Production-logs`, AWS menentukan kebijakan apa yang berlaku terhadap permintaan tersebut. Dalam hal ini, kebijakan berbasis identitas

yang dilampirkan ke pengguna `carlossalazar` adalah satu-satunya kebijakan yang berlaku di akun `111111111111`. Di akun `222222222222`, tidak ada kebijakan berbasis sumber daya yang dilampirkan ke bucket `Production-logs`. Saat AWS mengevaluasi akun `111111111111`, keputusan `Deny` dikembalikan. 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 [Menentukan apakah permintaan diizinkan atau ditolak dalam sebuah akun..](#)

Karena permintaan tersebut secara tegas ditolak di dalam salah satu akun, keputusan akhir adalah menolak permintaan tersebut.

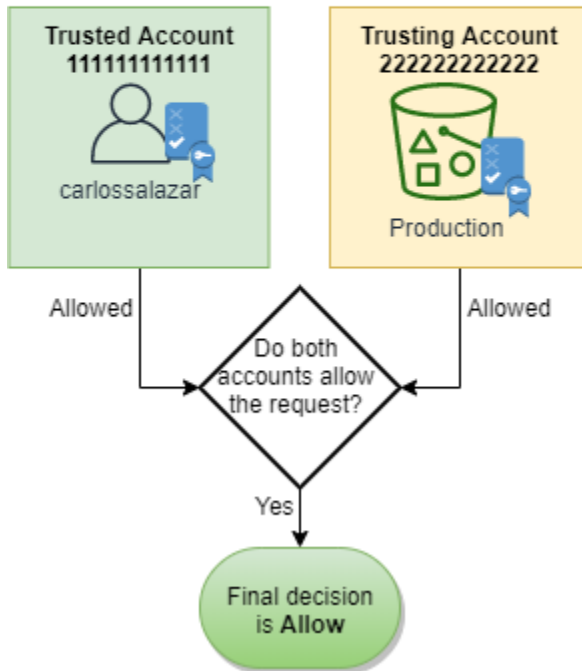


Asumsikan Carlos kemudian menyadari kesalahannya dan mencoba menyimpan file tersebut ke bucket `Production`. AWS pertama-tama memeriksa `111111111111` untuk menentukan apakah permintaan tersebut diizinkan. Hanya kebijakan berbasis identitas yang berlaku, dan ini memungkinkan permintaan tersebut. AWS selanjutnya 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.



Principal: carlossalazar
Action: s3:PutObject
Resource: Production



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 IAM](#)
- [Contoh kebijakan berbasis identitas IAM](#)
- [Contoh Kebijakan untuk Bekerja di Konsol Amazon EC2](#) dan [Contoh Kebijakan untuk Bekerja Dengan CLI, Amazon EC2 AWS CLI, atau SDK AWS di Panduan Pengguna Amazon EC2](#) untuk Instans Linux.

- [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 Anda lebih lanjut. Untuk mempelajari selengkapnya tentang validasi kebijakan, lihat [Memvalidasi kebijakan IAM](#). Untuk mempelajari selengkapnya 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 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>, ...])

<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 Prinsipal dalam Kebijakan di](#) Panduan Pengguna Layanan Penyimpanan 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 hanya karakter yang diperbolehkan dalam nilai Sid. Layanan AWS lain yang mendukung kebijakan sumber daya mungkin memiliki persyaratan lain untuk nilai Sid. Misalnya, beberapa layanan memerlukan nilai ini menjadi unik di dalam Akun AWS, dan beberapa layanan memungkinkan karakter tambahan seperti spasi dalam Sid nilai.

```
"Sid": "1"
```

```
"Sid": "ThisStatementProvidesPermissionsForConsoleAccess"
```

principal_id_string

Menyediakan cara untuk menentukan prinsipal menggunakan [Amazon Resource Name \(ARN\)](#) dari, pengguna IAM Akun AWS, peran IAM, pengguna federasi, atau pengguna peran yang dianggap. Untuk itu Akun AWS, Anda juga dapat menggunakan formulir pendek AWS: *accountnumber* alih-alih ARN penuh. Untuk semua opsi termasuk layanan AWS , peran yang diasumsikan, dan sebagainya, lihat [Menentukan prinsipal](#).

Perhatikan bahwa Anda dapat menggunakan * hanya untuk menentukan "semua orang/anonim". Anda tidak dapat menggunakannya luntuk menentukan bagian nama atau ARN.

resource_string

Dalam sebagian besar kasus, terdiri atas [Amazon Resource Name](#) (ARN)

```
"Resource": "arn:aws:iam::123456789012:user/Bob"

"Resource": "arn:aws:s3:::examplebucket/*"
```

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 IAM JSON: 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:PrincipalTypeaws:SecureTransport, danaws:useridd.

Untuk daftar kunci AWS kondisi, lihat [AWS kunci konteks kondisi global](#). Untuk kunci kondisi kepatuhan yang bersifat khusus terhadap layanan, lihat dokumentasi untuk layanan tersebut seperti berikut:

- [Menentukan Ketentuan dalam Kebijakan](#) di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon
- [Kebijakan IAM untuk Amazon EC2](#) di Panduan Pengguna Amazon EC2 untuk Instans Linux

```
"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 valid untuk jenis kondisi, lihat [Elemen kebijakan IAM JSON: 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 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. [AWSKebijakan yang dikelola](#) AWS kebijakan terkelola mencakup kasus penggunaan umum dan tersedia di Anda Akun AWS. 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 beralih ke izin hak istimewa terkecil, Anda dapat menjalankan AWS Identity and Access Management Access Analyzer untuk memantau prinsipal dengan kebijakan terkelola. AWS Setelah mengetahui izin yang mereka gunakan, 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 menggunakan AWS.

AWS kebijakan terkelola untuk fungsi pekerjaan dirancang untuk menyelaraskan secara dekat 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 `keiam: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 memungkinkan pengguna dengan kebijakan untuk meneruskan peran bernama "flow-logs-

vpc" ke CloudWatch layanan Amazon. 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 AWS Management Console. Di sana Anda dapat melihat dokumen kebijakan dan meninjau izin yang diberikan.

Fungsi pekerjaan administrator

AWS nama kebijakan terkelola: [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 kebijakan AWS 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](#) di Panduan Referensi Kebijakan AWS Terkelola.

Note

Sebelum pengguna atau peran IAM dapat mengakses AWS Billing and Cost Management konsol dengan izin dalam kebijakan ini, Anda harus terlebih dahulu mengaktifkan akses pengguna dan peran IAM. Untuk melakukannya, ikuti petunjuk di [Langkah 1 dari tutorial tentang pendelegasian akses ke konsol penagihan](#).

Fungsi pekerjaan penagihan

AWS nama kebijakan terkelola: [Penagihan](#)

Kasus penggunaan: Pengguna ini perlu melihat informasi penagihan, menyiapkan pembayaran, dan mengotorisasi pembayaran. Pengguna dapat memantau biaya yang terakumulasi 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 kebijakan AWS 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 Billing and Cost Management Pengguna. Untuk informasi selengkapnya tentang kebijakan terkelola, lihat [Penagihan](#) di Panduan Referensi Kebijakan AWS Terkelola.

Note

Sebelum pengguna atau peran IAM dapat mengakses AWS Billing and Cost Management konsol dengan izin dalam kebijakan ini, Anda harus terlebih dahulu mengaktifkan akses pengguna dan peran IAM. Untuk melakukannya, ikuti petunjuk di [Langkah 1 dari tutorial tentang pendelegasian akses ke konsol penagihan](#).

Fungsi pekerjaan administrator basis data

AWS nama kebijakan terkelola: [DatabaseAdministrator](#)

Kasus penggunaan: Pengguna ini menyiapkan, mengonfigurasi, dan memelihara database di Cloud. 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 tugas, lihat [Pembaruan kebijakan AWS 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](#) di Panduan Referensi Kebijakan AWS 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.

Peran layanan IAM opsional untuk fungsi kerja administrator basis data

Kasus penggunaan	Nama peran (* adalah wildcard)	Jenis peran layanan untuk dipilih	Pilih kebijakan AWS terkelola ini
Memungkinkan pengguna memantau basis data RDS	rds-monitoring-role	Peran Amazon RDS untuk Pemantauan yang Ditingkatkan	AmazonRDS EnhancedMonitoring Role
Memungkinkan AWS Lambda untuk memantau database Anda dan mengakses database eksternal	rdbms-lambda-access	Amazon EC2	AWSLambda_FullAccess
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-dinamodb-*	AWS Lambda	AWSLambda DynamoDBExecutionRole

Kasus penggunaan	Nama peran (* adalah wildcard)	Jenis peran layanan untuk dipilih	Pilih kebijakan AWS terkelola ini
Memungkinkan fungsi Lambda mengakses Amazon RDS dalam VPC	lambda-vpc-execution-role	Membuat peran dengan kebijakan kepercayaan seperti yang didefinisikan dalam Panduan AWS Lambda Pengembang	AWSLambdaVPCAccessExecutionRole
Izinkan AWS Data Pipeline untuk mengakses AWS sumber daya Anda	DataPipelineDefaultRole	Membuat peran dengan kebijakan kepercayaan seperti yang didefinisikan dalam Panduan AWS Data Pipeline Pengembang	AWS Data Pipeline Dokumentasi mencantumkan izin yang diperlukan untuk kasus penggunaan ini. Lihat peran IAM untuk AWS Data Pipeline
Izinkan aplikasi Anda berjalan di instans Amazon EC2 untuk mengakses sumber daya Anda AWS	DataPipelineDefaultResourceRole	Membuat peran dengan kebijakan kepercayaan seperti yang didefinisikan dalam Panduan AWS Data Pipeline Pengembang	AmazonEC2 RoleforDataPipelineRole

Fungsi pekerjaan ilmuwan data

AWS nama kebijakan terkelola: [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 kebijakan AWS terkelola untuk fungsi pekerjaan](#).

Deskripsi kebijakan: Kebijakan ini memberikan izin untuk membuat, mengelola, dan menjalankan kueri di kluster EMR Amazon dan melakukan analisis 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 Lihat kebijakan untuk mengetahui daftar lengkap layanan ilmuwan data yang mendukung kebijakan ini. Untuk informasi selengkapnya tentang kebijakan terkelola, lihat [DataScientist](#) di Panduan Referensi Kebijakan AWS Terkelola.

Kebijakan fungsi pekerjaan ini mendukung kemampuan untuk meneruskan peran ke AWS layanan. Satu pernyataan memungkinkan meneruskan peran apa pun ke SageMaker. 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.

Peran layanan IAM opsional untuk fungsi kerja ilmuwan data

Kasus penggunaan	Nama peran (* adalah wildcard)	Jenis peran layanan untuk dipilih	AWS kebijakan terkelola untuk memilih
Memungkinkan instans Amazon EC2 mengakses layanan dan sumber daya yang sesuai untuk kluster	EMR-EC2_DefaultRole	Amazon EMR for EC2	AmazonElasticMapReduceforEC2peran
Memungkinkan akses Amazon EMR untuk mengakses layanan dan	EMR_DefaultRole	Amazon EMR	ServicePolicyAmazonEMR_v2

Kasus penggunaan	Nama peran (* adalah wildcard)	Jenis peran layanan untuk dipilih	AWS kebijakan terkelola untuk memilih
sumber daya Amazon EC2 untuk klaster			
Izinkan Kinesis Managed Service untuk Apache Flink untuk 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 AWS Data Pipeline Pengembang	AWS Data Pipeline Dokumentasi mencantumkan izin yang diperlukan untuk kasus penggunaan ini. Lihat peran IAM untuk AWS Data Pipeline
Izinkan aplikasi Anda berjalan di instans Amazon EC2 untuk mengakses sumber daya Anda AWS	DataPipelineDefaultResourceRole	Membuat peran dengan kebijakan kepercayaan seperti yang didefinisikan dalam Panduan AWS Data Pipeline Pengembang	AmazonEC2RoleforDataPipelineRole

Fungsi pekerjaan pengguna daya developer

AWS nama kebijakan terkelola: [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 kebijakan AWS 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 izin kepada Organizations untuk melihat informasi tentang organisasi pengguna, termasuk email akun manajemen dan batasan organisasi. Meskipun kebijakan ini membatasi IAM, Organizations, ini memungkinkan pengguna untuk melakukan semua tindakan IAM Identity Center jika IAM Identity Center diaktifkan. Ini juga memberikan izin Manajemen Akun untuk melihat AWS Wilayah mana yang diaktifkan atau dinonaktifkan untuk akun tersebut.

Fungsi pekerjaan administrator jaringan

AWS nama kebijakan terkelola: [NetworkAdministrator](#)

Kasus penggunaan: Pengguna ini ditugaskan untuk menyiapkan 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 kebijakan AWS 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, Elastic CloudFront Load Balancing, Amazon SNS,, CloudWatch Log, AWS Elastic Beanstalk Amazon S3, IAM, dan Amazon CloudWatch Virtual Private Cloud. Untuk informasi selengkapnya tentang kebijakan terkelola, lihat [NetworkAdministrator](#) di Panduan Referensi Kebijakan AWS 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.

Peran layanan IAM opsional untuk fungsi kerja administrator jaringan

Kasus penggunaan	Nama peran (* adalah wildcard)	Jenis peran layanan untuk dipilih	AWS kebijakan terkelola 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	aliran-log-*	Membuat peran dengan kebijakan kepercayaan sebagaimana yang ditetapkan dalam Panduan Pengguna VPC Amazon	Kasus penggunaan ini tidak memiliki kebijakan AWS terkelola yang ada, tetapi dokumentasi mencantumkan izin yang diperlukan. Lihat Panduan Pengguna VPC Amazon .

Akses hanya-baca

AWS nama kebijakan terkelola: [ReadOnlyAccess](#)

Kasus penggunaan: Pengguna ini memerlukan akses hanya-baca ke setiap sumber daya dalam file. Akun 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 tugas, lihat [Pembaruan kebijakan AWS 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 hanya-baca ke AWS layanan terkait keamanan, seperti dan. AWS Identity and Access Management AWS Billing and Cost Management Lihat kebijakan untuk daftar lengkap layanan dan tindakan yang didukung kebijakan ini.

Fungsi pekerjaan auditor keamanan

AWS nama kebijakan terkelola: [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 kebijakan AWS terkelola untuk fungsi pekerjaan](#).

Deskripsi kebijakan: Kebijakan ini memberikan izin untuk melihat data konfigurasi untuk banyak AWS layanan dan meninjau log mereka. Untuk informasi selengkapnya tentang kebijakan terkelola, lihat [SecurityAudit](#) di Panduan Referensi Kebijakan AWS Terkelola.

Fungsi pekerjaan pengguna

AWS nama kebijakan terkelola: [SupportUser](#)

Kasus penggunaan: Pengguna ini menghubungi AWS Support, 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 kebijakan AWS terkelola untuk fungsi pekerjaan](#).

Deskripsi kebijakan: Kebijakan ini memberikan izin untuk membuat dan memperbarui AWS Support kasus. Untuk informasi selengkapnya tentang kebijakan terkelola, lihat [SupportUser](#) di Panduan Referensi Kebijakan AWS Terkelola.

Fungsi pekerjaan administrator sistem

AWS nama kebijakan terkelola: [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 kebijakan AWS terkelola untuk fungsi pekerjaan](#).

Deskripsi kebijakan: Kebijakan ini memberikan izin untuk membuat dan memelihara sumber daya di berbagai macam AWS layanan, termasuk, Amazon,,,,, CloudWatch, AWS CloudTrail Amazon EC2 AWS CodeCommit AWS CodeDeploy, AWS Config, AWS Directory Service, Amazon RDS AWS Identity and Access Management AWS Key Management Service, AWS Lambda Route 53, Amazon S3, Amazon SES, Amazon SQS,, dan Amazon VPC. AWS Trusted Advisor Untuk informasi selengkapnya tentang kebijakan terkelola, lihat [SystemAdministrator](#) di Panduan Referensi Kebijakan AWS 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 kebijakan AWS terkelola untuk fungsi pekerjaan](#).

Peran layanan IAM opsional untuk fungsi kerja administrator sistem

Kasus penggunaan	Nama peran (* adalah wildcard)	Jenis peran layanan untuk dipilih	AWS kebijakan terkelola untuk memilih
Memungkinkan aplikasi berjalan dalam instans EC2 di kluster Amazon ECS untuk mengakses Amazon ECS	ecr-sysadmin-*	Peran Amazon EC2 untuk Layanan EC2 Container	AmazonEC2EC2peranContainerServicefor
Memungkinkan pengguna untuk memantau basis data	rds-monitoring-role	Peran Amazon RDS untuk Pemantauan yang Ditingkatkan	AmazonRDSEnhancedMonitoringRole
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

Kasus penggunaan	Nama peran (* adalah wildcard)	Jenis peran layanan untuk dipilih	AWS kebijakan terkelola untuk memilih
			S3 seperti yang ditunjukkan dalam Panduan Pengguna Amazon EC2 untuk Instans Linux ; sesuaikan sesuai kebutuhan
Izinkan Lambda membaca aliran DynamoDB dan menulis ke Log CloudWatch	lambda-sysadmin-*	AWS Lambda	AWSLambda DynamoDBExecutionRole

Fungsi tugas pengguna hanya lihat

AWS nama kebijakan terkelola: [ViewOnlyAccess](#)

Kasus penggunaan: Pengguna ini dapat melihat daftar AWS sumber 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 kebijakan AWS terkelola untuk fungsi pekerjaan](#).

Deskripsi kebijakan: Kebijakan ini memberikan `List*`, `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](#) di Panduan Referensi Kebijakan AWS Terkelola.

Pembaruan kebijakan AWS 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 itu 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 [get-policy-version](#) AWS CLI perintah atau operasi [GetPolicyVersion](#) API.

Note

Anda dapat memiliki hingga lima versi kebijakan yang dikelola pelanggan, tetapi AWS mempertahankan riwayat versi lengkap kebijakan AWS terkelola.


Menciptakan peran dan memberlakukan kebijakan (konsol)

Beberapa kebijakan yang tercantum sebelumnya memberikan kemampuan untuk mengonfigurasi layanan AWS dengan peran yang memungkinkan layanan tersebut menjalankan 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 AWS Management Console 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 jenis entitas Tepercaya, pilih Layanan AWS.
4. Untuk kasus Layanan atau 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 Izin, opsi bergantung pada kasus penggunaan yang Anda pilih:
 - Jika layanan menentukan izin untuk peran tersebut, Anda tidak dapat memilih kebijakan izin.
 - Pilih dari serangkaian kebijakan izin terbatas.

- Pilih dari semua kebijakan izin.
 - Pilih kebijakan tanpa izin, buat kebijakan setelah peran dibuat, lalu lampirkan kebijakan ke peran.
7. (Opsional) Tetapkan [batas izin](#). Ini adalah fitur lanjutan yang tersedia untuk peran layanan, tetapi bukan peran tertaut layanan.
- a. Buka bagian **Setel 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 Peran, opsi bergantung 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.

 Important

Saat Anda memberi nama peran, perhatikan hal berikut:

- Nama peran harus unik di dalam diri AndaAkun AWS, dan tidak dapat dibuat unik berdasarkan kasus.

Misalnya, jangan membuat peran bernama keduanya **PRODRROLE** 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 tersebut 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 tersebut.
11. (Opsional) Untuk mengedit kasus penggunaan dan izin untuk peran, di 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 nilai kunci. Untuk informasi selengkapnya tentang penggunaan tanda di IAM, lihat [Menandai sumber daya IAM](#) 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 dipanggil `rds-monitoring-role` ke layanan Amazon RDS yang memungkinkan layanan untuk memantau database Amazon RDS atas nama mereka.

1. Masuk ke AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Pilih Kebijakan, **database** ketik kotak pencarian, lalu tekan enter.
3. Pilih tombol radio untuk DatabaseAdministratorkebijakan, pilih Tindakan, lalu pilih Lampirkan.
4. Dalam daftar entitas, pilih Alice dan kemudian pilih Lampirkan kebijakan. Alice kini dapat mengelola basis data AWS. 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 yang ditentukan oleh DatabaseAdministrator kebijakan yang dimiliki Alice sekarang. Salah satunya adalah **rds-monitoring-role**. Masukkan itu untuk nama Peran.
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 Aktifkan 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*` 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 AWS Management Console 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 Tindakan, lalu pilih Lampirkan.
4. Dalam daftar pengguna, pilih kotak centang di sebelah Jorge, lalu pilih Lampirkan 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.

```
{
  "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, ketik **vpc-flow-logs-policy-for-service-role** 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 AWSLayanan, lalu pilih Amazon EC2.
11. Pilih kasus penggunaan Amazon EC2.
12. Pada halaman Lampirkan kebijakan izin, pilih kebijakan yang Anda buat sebelumnya, vpc-flow-logs-policy-for-service-role, lalu pilih Berikutnya: Tinjau.
13. Nama peran harus diizinkan oleh NetworkAdministrator kebijakan yang dimiliki Jorge sekarang. Nama apa pun yang dimulai dengan flow-logs- diperbolehkan. Untuk contoh ini, masukkan **flow-logs-for-jorge** untuk 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 Peran, pilih flow-logs-for-jorgeperan (nama, bukan kotak centang). Pada halaman perincian peran baru Anda, pilih Hubungan kepercayaan, lalu pilih Edit hubungan kepercayaan.

- Ubah baris "Layanan" agar dibaca sebagai berikut, menggantikan entri untuk `ec2.amazonaws.com`:

```
"Service": "vpc-flow-logs.amazonaws.com"
```

- Jorge sekarang dapat membuat log aliran untuk VPC atau subnet di konsol Amazon EC2. Saat Anda membuat log alur, tentukan `flow-logs-for-jorge` perannya. Peran tersebut memiliki izin untuk membuat data log dan menuliskan data ke log itu.

AWS kunci konteks kondisi global

Ketika [kepala sekolah](#) membuat [permintaan](#) 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 permintaan dibuat terhadap, 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 [kepala AWS layanan](#). Untuk mempelajari lebih lanjut tentang keadaan di mana kunci global disertakan dalam konteks permintaan, lihat Informasi ketersediaan untuk setiap kunci.

Beberapa layanan individu membuat kunci kondisi mereka sendiri yang tersedia dalam konteks permintaan untuk layanan lain. Kunci kondisi lintas layanan adalah jenis kunci kondisi global yang menyertakan awalan yang cocok dengan nama layanan, seperti `ec2:atualambda:`, tetapi tersedia di seluruh layanan lain.

Kunci kondisi khusus layanan didefinisikan untuk digunakan dengan layanan individual AWS. 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 yang spesifik layanan, lihat [Tindakan, Sumber Daya, dan Kunci Kondisi untuk AWS Layanan dan pilih layanan](#) yang kuncinya ingin Anda lihat.

Note

Jika Anda menggunakan tombol kondisi yang hanya tersedia dalam beberapa keadaan, Anda dapat menggunakan [IfExists](#) versi 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.

```
"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 multivalued. Jangan gunakan operator set dengan kunci syarat bernilai tunggal. Untuk informasi selengkapnya, lihat [Kunci konteks multivaluasi](#).

Properti kepala sekolah	Properti sesi peran	Properti jaringan	Properti dari sumber daya	Properti permintaan
aws: Principal Arn	aws: Federated Provider	aws: SourceIp	aws: ResourceAccount	aws: CalledVia
aws: Principal Account	aws: TokenIssueTime	aws: SourceVpc	aws: ResourceOrgPaths	aws: CalledVia First
aws: Principal OrgPaths	aws: MultiFactorAuthAge	aws: VpcSourceIp	aws: ResourceOrg ID	aws: CalledVia Last
				AWS: melalui AWSService

Properti kepala sekolah	Properti sesi peran	Properti jaringan	Properti dari sumber daya	Properti permintaan
aws: Principal Org ID	aws: MultiFactorAuthPresent		aws:ResourceTag/tag-kunci	aws: CurrentTime
aws:PrincipalTag/tag-kunci	AWS: EC2 InstanceSourceVpc			aws: EpochTime
aws: Principalls AWSService	AWS: EC2 IPv4 InstanceSourcePrivate			aws:referer
aws: Principal ServiceName	aws: SourceIdentity			aws: Requested Region
aws: Principal ServiceNamesList	EC2: RoleDelivery			aws: RequestTag/tag-kunci
aws: Principal Type	EC2: SourceInstanceArn			aws: TagKeys
aws:userid	Iam: RoleAssumedBy			aws: SecureTransport
aws:username	Iam: CredentialIssuingService			aws: SourceArn
	lambda: SourceFunctionArn			aws: SourceAccount
	ssm: SourceInstanceArn			aws: SourceOrg Paths
	toko identitas: UserId			aws: SourceOrg ID
				aws: UserAgent

Properti kepala sekolah

Gunakan tombol 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, lihat. [Menentukan prinsipal](#)

Daftar Isi

- [aws: PrincipalArn](#)
- [aws: PrincipalAccount](#)
- [aws: PrincipalOrgPaths](#)
- [aws: PrincipalOrg ID](#)
- [aws:PrincipalTag/tag-kunci](#)
- [aws: Principals AWSService](#)
- [aws: PrincipalServiceName](#)
- [aws: PrincipalServiceNamesList](#)
- [aws: PrincipalType](#)
- [aws:userid](#)
- [aws:username](#)

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:
 - Peran IAM
 - Pengguna IAM
 - AWS STS sesi pengguna federasi
 - Akun AWS pengguna root
- Tipe data - ARN, String

AWS merekomendasikan agar Anda menggunakan operator [ARN alih-alih operator string](#) saat membandingkan ARN.

- Jenis nilai - Bernilai tunggal
- Contoh nilai Daftar berikut menunjukkan nilai konteks permintaan yang dikembalikan untuk berbagai jenis prinsipal yang dapat Anda tentukan dalam kunci kondisi: `aws:PrincipalArn`
- Peran IAM - Konteks permintaan berisi nilai berikut untuk kunci `aws:PrincipalArn` kondisi. Jangan tentukan ARN sesi peran yang diasumsikan sebagai nilai untuk kunci kondisi ini. Untuk informasi lebih lanjut tentang kepala sesi peran yang diasumsikan, lihat [Kepala 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 sesi pengguna federasi - Konteks permintaan berisi nilai berikut untuk kunci `aws:PrincipalArn` kondisi.

```
arn:aws:sts::123456789012:federated-user/user-name
```

- Akun AWS root user - Konteks permintaan berisi nilai berikut untuk kunci kondisi `aws:PrincipalArn`. Saat Anda menentukan ARN pengguna root sebagai nilai untuk kunci `aws:PrincipalArn` kondisi, itu membatasi izin hanya untuk pengguna root. Akun AWS ini berbeda dengan menentukan ARN pengguna root dalam elemen utama kebijakan berbasis sumber daya, yang mendelegasikan wewenang ke. Akun AWS Untuk informasi selengkapnya tentang menentukan ARN pengguna root di elemen utama kebijakan berbasis sumber daya, lihat [Akun AWS kepala sekolah](#)

```
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 Pengguna Organizations.

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 akan ditampilkan anonymous.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan untuk semua permintaan, termasuk permintaan anonim.
- Tipe data - [String](#)
- Jenis nilai - Bernilai tunggal

Dalam contoh berikut, akses ditolak kecuali ke kepala sekolah dengan nomor rekening. 123456789012

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyAccessFromPrincipalNotInSpecificAccount",
      "Action": "service:*",
      "Effect": "Deny",
      "Resource": [
        "arn:aws:service:region:accountID:resource"
      ],
      "Condition": {
        "StringNotEquals": {
          "aws:PrincipalAccount": [
            "123456789012"
          ]
        }
      }
    }
  ]
}
```

aws: PrincipalOrgPaths

Gunakan kunci ini untuk membandingkan AWS Organizations jalur bagi kepala sekolah yang membuat permintaan ke jalur dalam kebijakan. Prinsipal itu dapat berupa pengguna IAM, peran IAM, pengguna 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. AWS Organizations Path adalah representasi teks dari struktur entitas Organizations. Untuk informasi lebih lanjut tentang cara menggunakan dan memahami jalur, lihat [Memahami jalur entitas AWS Organizations](#).

- 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 — Multivalued

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 mengembalikan `true` prinsipal di akun yang dilampirkan langsung ke `ou-ab12-22222222` OU, tetapi tidak pada OU anaknya.

```
"Condition" : { "ForAnyValue:StringEquals" : {  
    "aws:PrincipalOrgPaths":["o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/"]  
  }  
}}
```

Kondisi berikut kembali `true` untuk prinsipal dalam akun yang dilampirkan langsung ke OU atau salah satu OU anaknya. 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 kembali `true` untuk prinsipal dalam akun yang dilampirkan langsung ke salah satu OU anak, tetapi tidak langsung ke OU orang tua. 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 multivalai. Kunci multivalued 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 multivalai ini, lihat [Kunci konteks multivaluasi](#).

```
"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 pengenalan organisasi tempat prinsipal permintaan berada dengan pengenalan yang ditentukan dalam kebijakan. AWS Organizations

- **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 - Bernilai tunggal

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-xxxxxxxxxxxx` untuk menambahkan objek ke dalam bucket `policy-ninja-dev`.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "AllowPutObject",
    "Effect": "Allow",
    "Principal": "*",
    "Action": "s3:PutObject",
    "Resource": "arn:aws:s3:::policy-ninja-dev/*",
    "Condition": {"StringEquals":
      {"aws:PrincipalOrgID": "o-xxxxxxxxxxxx"}
    }
  }
}
```

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, lihat [aws: Principalls AWSService](#).

Untuk informasi lebih lanjut tentang AWS Organizations, lihat [Apa itu AWS Organizations?](#) dalam AWS Organizations User Guide.

aws:PrincipalTag/tag-kunci

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 - Bernilai tunggal

Anda dapat menambahkan atribut khusus ke pengguna atau peran dalam bentuk pasangan nilai kunci. Untuk informasi selengkapnya tentang tag IAM, lihat [Menandai sumber daya IAM](#). 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 *teks placeholder yang dicetak miring* dalam kebijakan contoh dengan informasi Anda sendiri. Lalu, ikuti petunjuk di [buat kebijakan](#) atau [ubah kebijakan](#).

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iam:*",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:PrincipalTag/department": "hr"
        }
      }
    }
  ]
}
```

aws: Principals AWSService

Gunakan kunci ini untuk memeriksa apakah panggilan ke sumber daya Anda dilakukan langsung oleh [kepala AWS 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 kredensi prinsipal IAM untuk membuat permintaan atas nama kepala sekolah. Itu juga diatur ke salah jika layanan menggunakan [peran layanan](#) atau [peran tertaut layanan](#) untuk melakukan panggilan atas nama prinsipal.

- 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 - Bernilai tunggal

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 tersebut berasal dari `vpc-111bbb22` atau berasal dari prinsipal layanan, seperti. CloudTrail

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Expected-network+service-principal",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::DOC-EXAMPLE-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 secara aman ke pengguna resmi Anda, lokasi jaringan yang diharapkan, dan AWS layanan secara bersamaan.](#)

`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 ada dalam permintaan saat panggilan dilakukan oleh kepala AWS layanan. Kunci ini tidak ada dalam situasi lain, termasuk berikut ini:
 - Jika layanan menggunakan [peran layanan](#) atau [peran tertaut layanan](#) untuk melakukan panggilan atas nama prinsipal.
 - Jika layanan menggunakan kredensi kepala IAM untuk mengajukan permintaan atas nama kepala sekolah.
 - Jika panggilan dilakukan langsung oleh kepala sekolah IAM.
 - Jika panggilan dilakukan oleh pemohon anonim.
- Tipe data - [String](#)
- Jenis nilai - Bernilai tunggal

Anda dapat menggunakan kunci kondisi ini untuk membatasi akses ke identitas tepercaya dan lokasi jaringan yang diharapkan sambil memberikan akses ke layanan dengan aman. AWS

Dalam contoh kebijakan bucket Amazon S3 berikut, akses ke bucket dibatasi kecuali permintaan tersebut berasal dari `vpc-111bbb22` atau berasal dari prinsipal layanan, seperti. CloudTrail

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

```
"Statement": [  
  {  
    "Sid": "expected-network+service-principal",  
    "Effect": "Deny",  
    "Principal": "*",  
    "Action": "s3:PutObject",  
    "Resource": "arn:aws:s3:::DOC-EXAMPLE-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 dapat membuat prinsipal layanan Regional untuk menunjukkan instans layanan tertentu 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 ada dalam permintaan saat panggilan dilakukan oleh kepala AWS layanan. Kunci ini tidak ada dalam situasi lain, termasuk berikut ini:
 - Jika layanan menggunakan [peran layanan](#) atau [peran tertaut layanan](#) untuk melakukan panggilan atas nama prinsipal.
 - Jika layanan menggunakan kredensi kepala IAM untuk mengajukan 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 — Multivalued

`aws:PrincipalServiceNamesList` adalah kunci kondisi multinilai. Kunci multivalued dapat memiliki beberapa nilai dalam konteks permintaan. Anda harus menggunakan `ForAnyValue` atau `ForAllValues` mengatur operator dengan [operator kondisi string](#) untuk kunci ini. Untuk informasi selengkapnya tentang kunci kondisi multinilai ini, lihat [Kunci konteks multivaluasi](#).

`aws: PrincipalType`

Gunakan kunci ini untuk membandingkan tipe prinsipal yang membuat permintaan dengan tipe prinsipal yang Anda sebutkan dalam kebijakan. Untuk informasi selengkapnya, lihat [Menentukan prinsipal](#). Untuk contoh spesifik dari nilai-nilai `principal` kunci, lihat [Nilai-nilai kunci utama](#).

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan untuk semua permintaan, termasuk permintaan anonim.
- Tipe data - [String](#)
- Jenis nilai - Bernilai tunggal

`aws:userid`

Gunakan kunci ini untuk membandingkan pengenalan 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 [Menentukan prinsipal](#). Untuk contoh spesifik dari nilai-nilai `principal` kunci, lihat [Nilai-nilai kunci utama](#).

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan untuk semua permintaan, termasuk permintaan anonim.
- Tipe data - [String](#)
- Jenis nilai - Bernilai tunggal

`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 [Menentukan prinsipal](#). Untuk contoh spesifik dari nilai-nilai `principal` kunci, lihat [Nilai-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 kredensial IAM Identity Center tidak menyertakan kunci ini dalam konteksnya.

- Tipe data - [String](#)
- Jenis nilai - Bernilai tunggal

Properti sesi peran

Gunakan tombol kondisi berikut untuk membandingkan properti sesi peran pada saat sesi dibuat. Kunci kondisi ini hanya tersedia jika permintaan dibuat oleh prinsipal dengan sesi peran atau kredensial pengguna gabungan. Nilai untuk kunci kondisi ini disematkan dalam token sesi peran.

[Peran](#) adalah jenis kepala sekolah. Anda juga dapat menggunakan kunci kondisi dari [Properti kepala sekolah](#) bagian untuk mengevaluasi properti peran saat peran membuat permintaan.

Daftar Isi

- [aws: FederatedProvider](#)
- [aws: TokenIssueTime](#)
- [aws: MultiFactorAuthAge](#)
- [aws: MultiFactorAuthPresent](#)
- [AWS: EC2 InstanceSourceVpc](#)
- [AWS: EC2 IPv4 InstanceSourcePrivate](#)
- [aws: SourceIdentity](#)
- [EC2: RoleDelivery](#)
- [EC2: SourceInstanceArn](#)
- [lem: RoleAssumedBy](#)
- [lem: CredentialIssuingService](#)
- [lambda: SourceFunctionArn](#)
- [ssm: SourceInstanceArn](#)
- [toko identitas: UserId](#)

aws: FederatedProvider

Gunakan kunci ini untuk membandingkan penyedia identitas penerbit (iDP) prinsipal dengan iDP yang Anda tentukan dalam kebijakan. Ini berarti bahwa peran IAM diasumsikan menggunakan

`AssumeRoleWithWebIdentity` AWS STS operasi. Bila kredensi sementara sesi peran yang dihasilkan digunakan untuk membuat permintaan, konteks permintaan mengidentifikasi IDP yang mengautentikasi identitas federasi asli.

- Ketersediaan — Kunci ini hadir ketika prinsipal adalah kepala sesi peran dan sesi itu dikeluarkan ketika peran diasumsikan `AssumeRoleWithWebIdentity`.
- Tipe data - [String](#)
- Jenis nilai - Bernilai tunggal

Misalnya, jika pengguna diautentikasi melalui Amazon Cognito, konteks permintaan menyertakan nilainya `cognito-identity.amazonaws.com`. Demikian pula, jika pengguna diautentikasi melalui Login with Amazon, konteks permintaan menyertakan nilainya `www.amazon.com`.

Anda dapat menggunakan kunci kondisi bernilai tunggal apa pun sebagai [variabel](#). Contoh kebijakan berbasis sumber daya berikut menggunakan `aws:FederatedProvider` kunci sebagai variabel kebijakan di ARN sumber daya. Kebijakan ini memungkinkan setiap prinsipal yang melakukan autentikasi menggunakan iDP untuk mengeluarkan objek dari bucket Amazon S3 dengan jalur yang khusus untuk penyedia identitas penerbit.

`aws: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. Kuncinya tidak ada dalam AWS CLI, AWS API, atau permintaan AWS SDK yang dibuat menggunakan kunci akses.
- Tipe data - [Tanggal](#)
- Jenis nilai - Bernilai tunggal

Untuk mempelajari layanan mana yang mendukung penggunaan kredensial sementara, lihat [AWS layanan yang bekerja dengan IAM](#).

`aws: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 [Menggunakan otentikasi multi-faktor \(MFA\) di AWS](#).

⚠ Important

Kunci kondisi ini tidak ada untuk identitas gabungan atau permintaan yang dibuat menggunakan kunci akses untuk menandatangani permintaan AWS CLI, AWS API, atau SDK. AWS Untuk mempelajari selengkapnya tentang menambahkan perlindungan MFA ke operasi API dengan kredensial keamanan sementara, lihat. [Mengonfigurasi akses API yang dilindungi MFA](#)

Untuk memeriksa apakah MFA digunakan untuk memvalidasi identitas federasi IAM, Anda dapat meneruskan metode otentikasi dari penyedia identitas Anda ke tag sesi. AWS Untuk detailnya, lihat [Melewati tag sesi di AWS STS](#). Untuk menerapkan MFA untuk identitas Pusat Identitas IAM, Anda [dapat mengaktifkan atribut untuk kontrol akses untuk](#) meneruskan klaim pernyataan SAFL dengan metode otentikasi dari penyedia identitas Anda ke IAM Identity Center.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan hanya jika prinsipal menggunakan [kredensial keamanan sementara](#) untuk membuat permintaan. Kebijakan dengan kondisi MFA dapat dilampirkan pada:
 - 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 - Bernilai tunggal

aws: MultiFactorAuthPresent

Gunakan kunci ini untuk memeriksa apakah otentikasi multi-faktor (MFA) digunakan untuk memvalidasi [kredensial keamanan sementara](#) yang membuat permintaan.

⚠ Important

Kunci kondisi ini tidak ada untuk identitas gabungan atau permintaan yang dibuat menggunakan kunci akses untuk menandatangani permintaan AWS CLI, AWS API, atau SDK. AWS Untuk mempelajari selengkapnya tentang menambahkan perlindungan MFA ke

operasi API dengan kredensial keamanan sementara, lihat. [Mengonfigurasi akses API yang dilindungi MFA](#)

Untuk memeriksa apakah MFA digunakan untuk memvalidasi identitas federasi IAM, Anda dapat meneruskan metode otentikasi dari penyedia identitas Anda ke tag sesi. AWS Untuk detailnya, lihat [Melewati tag sesi di AWS STS](#). Untuk menerapkan MFA untuk identitas Pusat Identitas IAM, Anda [dapat mengaktifkan atribut untuk kontrol akses untuk](#) meneruskan klaim pernyataan SAFL dengan metode otentikasi dari penyedia identitas Anda ke IAM Identity Center.

- Ketersediaan – Kunci ini disertakan dalam konteks permintaan hanya jika prinsipal menggunakan kredensial sementara untuk membuat permintaan. Kebijakan dengan kondisi MFA dapat dilampirkan pada:
 - 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 - Bernilai tunggal

Kredensial sementara digunakan untuk mengautentikasi peran IAM dan pengguna IAM dengan token sementara dari [AssumeRole](#) atau [GetSessionToken](#), dan pengguna. AWS Management Console

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 AWS Management Console 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 kredensial pengguna untuk membuat permintaan lain ke layanan lain. Misalnya, saat memanggil Athena untuk mengakses bucket Amazon S3, atau saat AWS CloudFormation menggunakan

untuk membuat instans Amazon EC2. Untuk permintaan berikutnya, AWS gunakan kredensial 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 otentikasi-MFA 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 terautentikasi 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 menyangkal 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 CLI atau AWS API.

Kombinasi Alternatif

Anda juga dapat menggunakan [BoolIfExists](#) operator untuk mengizinkan permintaan yang diautentikasi MFA dan AWS CLI 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. Artinya AWS CLI, operasi AWS API, dan AWS SDK diizinkan saat 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 dari `Allow`, `Bool`, dan `true` hanya mengizinkan permintaan yang diautentikasi oleh MFA. 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 autentikasi MFA benar-benar digunakan.

AWS: EC2 InstanceSourceVpc

Kunci ini mengidentifikasi VPC tempat kredensial peran Amazon EC2 IAM dikirimkan. Anda dapat menggunakan kunci ini dalam kebijakan dengan kunci [aws:SourceVPC](#) global untuk memeriksa apakah panggilan dilakukan dari VPC (`aws:SourceVPC`) yang cocok dengan VPC tempat kredensi 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 - Bernilai tunggal

Kunci ini dapat digunakan dengan nilai pengidentifikasi VPC, tetapi paling berguna bila digunakan sebagai variabel yang dikombinasikan dengan kunci konteks. `aws:SourceVpc` Kunci `aws:SourceVpc` 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 yang biasanya berubah bersama.

Note

Kunci kondisi ini tidak tersedia di EC2-Classik.

```
{
```

```
"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"
      }
    }
  }
]
```

Pada contoh di atas, akses ditolak jika `aws:SourceVpc` nilainya tidak sama dengan `aws:Ec2InstanceSourceVpc` nilainya. 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` AWS untuk mengizinkan otorisasi permintaan 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 panggilan atas nama Anda. AWS KMS Beberapa kunci tidak ada saat permintaan dibuat AWS KMS.

AWS: EC2 IPv4 InstanceSourcePrivate

Kunci ini mengidentifikasi alamat IPv4 pribadi dari antarmuka elastic network primer yang dikirimkan kredensial peran Amazon EC2 IAM. Anda harus menggunakan kunci kondisi ini dengan kunci pendampingnya `aws:Ec2InstanceSourceVpc` untuk memastikan bahwa Anda memiliki kombinasi unik global 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 kredensialnya.

- 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 - Bernilai tunggal

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 Anda menggunakan `aws:Ec2InstanceSourcePrivateIPv4` kunci untuk menentukan VPC dari kredensial instans Amazon EC2 Anda.

Note

Kunci kondisi ini tidak tersedia di EC2-Classic.

```
{
  "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 saat peran diasumsikan menggunakan perintah CLI AWS STS peran apa pun, atau operasi API. `AWS STS AssumeRole`
- Tipe data - [String](#)
- Jenis nilai - Bernilai tunggal

Anda dapat menggunakan kunci ini dalam kebijakan untuk mengizinkan tindakan AWS berdasarkan prinsipal yang telah menetapkan identitas sumber saat mengambil peran. Aktivitas untuk identitas sumber yang ditentukan peran muncul di [AWS CloudTrail](#). Hal ini memudahkan administrator untuk menentukan siapa atau apa yang melakukan tindakan dengan peran dalam 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 kredensial sesi untuk mengasumsikan peran lain. Mengasumsikan satu peran dari peran lain disebut [rantai peran](#).

[sts:SourceIdentity](#) Kuncinya ada dalam permintaan ketika prinsipal awalnya menetapkan identitas sumber sambil mengasumsikan peran menggunakan perintah CLI AWS STS peran apa pun, atau operasi API. AWS STS AssumeRole 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.

```
{
  "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 kredensial 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 - Bernilai tunggal
- 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) - Metode permintaan/respons
- Layanan Metadata Instans Versi 2 (IMDSv2) - metode berorientasi sesi

Untuk informasi tentang cara mengonfigurasi instans Anda agar menggunakan IMDSv2, lihat [Mengonfigurasi 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 kredensial peran Amazon EC2, maka tidak akan berpengaruh.

```
{
  "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 instance](#).

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 - Bernilai tunggal
- Nilai contoh — `arn:aws:ec2:us-west - 2:111111111111: instance/instance-id`

Untuk contoh kebijakan, lihat [Mengizinkan instance tertentu untuk melihat sumber daya di AWS layanan lain](#).

Iem: RoleAssumedBy

AWS Glue Layanan menetapkan kunci kondisi ini untuk setiap permintaan AWS API yang 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 - Bernilai tunggal
- Nilai contoh - Kunci ini selalu diatur ke `glue.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:::confidential-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 - Bernilai tunggal
- Nilai contoh - Kunci ini selalu diatur ke `glue.amazonaws.com`.

Contoh berikut menambahkan kondisi yang melekat pada peran IAM yang digunakan oleh AWS Glue pekerjaan. Ini memastikan tindakan tertentu diizinkan/ditolak berdasarkan apakah sesi peran digunakan untuk lingkungan runtime AWS Glue pekerjaan.

```
{
  "Effect": "Allow",
  "Action": "s3:GetObject",
  "Resource": "arn:aws:s3:::confidential-bucket/*",
  "Condition": {
    "StringEquals": {
      "glue:CredentialIssuingService": "glue.amazonaws.com"
    }
  }
}
```

lambda: SourceFunctionArn

Gunakan kunci ini untuk mengidentifikasi ARN fungsi Lambda yang dikirimkan kepada kredensial peran IAM. 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 - Bernilai tunggal
- Nilai contoh - arn:aws:lambda: us-east- 1:123456789012: fungsi: TestFunction

Contoh berikut memungkinkan satu fungsi Lambda tertentu untuk s3:PutObject mengakses bucket yang ditentukan.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ExampleSourceFunctionArn",
      "Effect": "Allow",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*",
      "Condition": {
        "ArnEquals": {
          "lambda:SourceFunctionArn": "arn:aws:lambda:us-
east-1:123456789012:function:source_lambda"
        }
      }
    }
  ]
}
```

Untuk informasi selengkapnya, lihat [Bekerja dengan kredensial lingkungan eksekusi Lambda](#) di Panduan Pengembang.AWS Lambda

ssm: SourceInstanceArn

Gunakan kunci ini untuk mengidentifikasi ARN instance AWS Systems Manager terkelola yang dikirimkan ke kredensial peran IAM. Kunci kondisi ini tidak ada saat permintaan berasal dari instance terkelola dengan peran IAM yang terkait dengan profil instans Amazon EC2.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan setiap kali kredensial peran dikirimkan ke instance AWS Systems Manager terkelola.
- Tipe data — [ARN](#)
- Jenis nilai - Bernilai tunggal
- Nilai contoh — arn:aws:ec2:us-west - 2:11111111111: instance/instance-id

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 - Bernilai tunggal
- Nilai contoh — 94482488-3041-7026-18f3-be45837cd0e4

Anda dapat menemukan pengguna di IAM Identity Center dengan membuat permintaan ke [GetUserId](#) API menggunakan API AWS CLI, AWS API, atau AWS SDK. UserId

Properti jaringan

Gunakan tombol kondisi berikut untuk membandingkan detail tentang jaringan tempat permintaan berasal atau diteruskan dengan properti jaringan yang Anda tentukan dalam kebijakan.

Daftar Isi


- [aws: SourceIp](#)
- [aws: SourceVpc](#)
- [aws: SourceVpce](#)
- [aws: VpcSourceIp](#)

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.
- Tipe data — [alamat IP](#)
- Jenis nilai - Bernilai tunggal

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 dukungan IPv6 Layanan AWS itu, lihat [Layanan AWS dukungan 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 `DOC-EXAMPLE-BUCKET3` Amazon S3 jika mereka melakukan panggilan dari rentang alamat IPv4 yang ditentukan. Kebijakan ini juga memungkinkan AWS layanan yang digunakan [Teruskan sesi akses](#) untuk melakukan operasi ini atas nama Anda.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PrincipalPutObjectIfIpAddress",
      "Effect": "Allow",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::DOC-EXAMPLE-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 `DOC-EXAMPLE-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 Layanan AWS Anda bekerja dengan mendukung IPv6. Untuk

daftar dukungan IPv6 Layanan AWS itu, lihat [Layanan AWS dukungan IPv6 di Panduan Pengguna Amazon VPC](#).

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PrincipalPutObjectIfIpAddress",
      "Effect": "Allow",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::DOC-EXAMPLE-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 kunci khusus VPC seperti `aws:VpcSourceIp`](#) Untuk informasi selengkapnya tentang penggunaan titik akhir VPC, lihat [Identitas dan manajemen akses untuk titik akhir VPC dan layanan titik akhir VPC](#) di Panduan.AWS PrivateLink

`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 - Bernilai tunggal

aws: SourceVpce

Gunakan kunci ini untuk membandingkan pengidentifikasi VPC endpoint dari permintaan dengan ID titik akhir yang Anda sebutkan dalam kebijakan. Dalam kebijakan, Anda dapat menggunakan kunci ini untuk membatasi akses ke VPC endpoint tertentu. Untuk informasi selengkapnya, lihat [Membatasi Akses ke Titik Akhir 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 - Bernilai tunggal

aws: 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.
- Tipe data — [alamat IP](#)
- Jenis nilai - Bernilai tunggal

Untuk informasi selengkapnya, silakan lihat [Mengendalikan Akses ke Layanan dengan titik akhir VPC](#) dalam Panduan Pengguna Amazon VPC.

Note

`aws:VpcSourceIp` mendukung alamat IPv4 dan IPv6 atau rentang alamat IP. Untuk daftar dukungan IPv6 Layanan AWS itu, lihat [Layanan AWS dukungan IPv6 di Panduan Pengguna Amazon VPC](#).

Properti dari sumber daya

Gunakan tombol kondisi berikut untuk membandingkan detail tentang sumber daya yang menjadi target permintaan dengan properti sumber daya yang Anda tentukan dalam kebijakan.

Daftar Isi

- [aws: ResourceAccount](#)
- [aws: ResourceOrgPaths](#)
- [aws: ResourceOrg ID](#)
- [aws:ResourceTag/tag-kunci](#)

aws: 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`
 - Amazon Elastic Block Store - Semua tindakan
 - Amazon EC2
 - `ec2:AcceptTransitGatewayPeeringAttachment`
 - `ec2:AcceptVpcEndpointConnections`
 - `ec2:AcceptVpcPeeringConnection`
 - `ec2:CopyFpgaImage`
 - `ec2:CopyImage`
 - `ec2:CopySnapshot`
 - `ec2:CreateTransitGatewayPeeringAttachment`
 - `ec2:CreateVolume`
 - `ec2:CreateVpcEndpoint`

- `ec2:CreateVpcPeeringConnection`
- `ec2>DeleteTransitGatewayPeeringAttachment`
- `ec2>DeleteVpcPeeringConnection`
- `ec2:RejectTransitGatewayPeeringAttachment`
- `ec2:RejectVpcEndpointConnections`
- `ec2:RejectVpcPeeringConnection`
- Amazon EventBridge
 - `events:PutEvents`— EventBridge `PutEvents` memanggil 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 peristiwa dari AWS akun lain](#) di Panduan EventBridge Pengguna Amazon.
- Amazon GuardDuty
 - `guardduty:AcceptAdministratorInvitation`
- Amazon Macie
 - `macie2:AcceptInvitation`
- Amazon Route 53
 - `route53:AssociateVpcWithHostedZone`
 - `route53>CreateVPCAssociationAuthorization`
 - `route53>DeleteVPCAssociationAuthorization`
 - `route53:DisassociateVPCFromHostedZone`
 - `route53:ListHostedZonesByVPC`
- AWS Security Hub
 - `securityhub:AcceptAdministratorInvitation`
- Amazon WorkSpaces
 - `workspaces:DescribeWorkspaceImages`
- Tipe data - [String](#)
- Jenis nilai - Bernilai tunggal

 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:::DOC-EXAMPLE-BUCKET2`- Bucket Amazon S3 dibuat dan dimiliki di dalam akun111122223333, 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 placeholder yang dicetak 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.

```
{
```

```
"Version": "2012-10-17",
"Statement": [
  {
    "Sid": "DenyInteractionWithResourcesNotInSpecificAccount",
    "Action": "service:*",
    "Effect": "Deny",
    "Resource": [
      "arn:aws:service:region:account:*"
    ],
    "Condition": {
      "StringNotEquals": {
        "aws:ResourceAccount": [
          "account"
        ]
      }
    }
  }
]
```

Kebijakan ini menolak akses ke semua sumber daya untuk AWS layanan tertentu kecuali yang ditentukan Akun AWS memiliki sumber daya.

Note

Beberapa Layanan AWS memerlukan akses ke sumber daya yang AWS dimiliki yang di-host 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, mengandalkan akses ke sumber daya di luar Anda Akun AWS untuk operasi normal. Jika Anda menggunakan elemen `aws:ResourceAccount` 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 menentukan pengecualian untuk sumber daya milik layanan.


Gunakan contoh kebijakan ini sebagai templat untuk membuat kebijakan kustom Anda sendiri. Lihat [dokumentasi](#) layanan Anda untuk informasi lebih lanjut.

aws: ResourceOrgPaths

Gunakan kunci ini untuk membandingkan jalur AWS Organizations untuk 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. AWS Organizations path adalah representasi teks dari struktur entitas Organizations. Untuk informasi selengkapnya tentang menggunakan dan memahami jalur, lihat [Memahami jalur entitas AWS Organizations](#)

- 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`
 - Amazon Elastic Block Store - Semua tindakan
 - Amazon EC2
 - `ec2:AcceptTransitGatewayPeeringAttachment`
 - `ec2:AcceptVpcEndpointConnections`
 - `ec2:AcceptVpcPeeringConnection`
 - `ec2:CopyFpgaImage`
 - `ec2:CopyImage`
 - `ec2:CopySnapshot`
 - `ec2:CreateTransitGatewayPeeringAttachment`
 - `ec2:CreateVolume`
 - `ec2:CreateVpcEndpoint`
 - `ec2:CreateVpcPeeringConnection`
 - `ec2>DeleteTransitGatewayPeeringAttachment`
 - `ec2>DeleteVpcPeeringConnection`
 - `ec2:RejectTransitGatewayPeeringAttachment`
 - `ec2:RejectVpcEndpointConnections`
 - `ec2:RejectVpcPeeringConnection`
 - Amazon EventBridge

- `events:PutEvents`— EventBridge `PutEvents` memanggil 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 peristiwa dari AWS akun lain](#) di Panduan EventBridge Pengguna Amazon.
- Amazon GuardDuty
 - `guardduty:AcceptAdministratorInvitation`
- Amazon Macie
 - `macie2:AcceptInvitation`
- Amazon Route 53
 - `route53:AssociateVpcWithHostedZone`
 - `route53:CreateVPCAssociationAuthorization`
 - `route53>DeleteVPCAssociationAuthorization`
 - `route53:DisassociateVPCFromHostedZone`
 - `route53:ListHostedZonesByVPC`
- AWS Security Hub
 - `securityhub:AcceptAdministratorInvitation`
- Amazon WorkSpaces
 - `workspaces:DescribeWorkspaceImages`
- Tipe data - [String](#) (daftar)
- Jenis nilai — Multivalued

 Note

Untuk pertimbangan tambahan untuk tindakan yang tidak didukung di atas, lihat repositori [Contoh Kebijakan Perimeter Data](#).

`aws:ResourceOrgPaths` adalah kunci kondisi multivalued. Kunci multivalued dapat memiliki beberapa nilai dalam konteks permintaan. Anda harus menggunakan `ForAnyValue` atau `ForAllValues` mengatur operator dengan [operator kondisi string](#) untuk kunci ini. Untuk informasi selengkapnya tentang kunci kondisi multivalued ini, lihat [Kunci konteks multivalued](#).

Misalnya, kondisi berikut mengembalikan `True` sumber daya milik organisasi-`a1b2c3d4e5`. Ketika Anda menyertakan wildcard, Anda harus menggunakan operator [StringLike](#) kondisi.

```
"Condition": {
  "ForAnyValue:StringLike": {
    "aws:ResourceOrgPaths":["o-a1b2c3d4e5/*"]
  }
}
```

Kondisi berikut mengembalikan `True` sumber daya dengan ID `OUou-ab12-11111111`. Ini akan cocok dengan sumber daya yang dimiliki oleh akun yang dilampirkan ke OU `ou-ab12-11111111` atau salah satu dari OU anak.

```
"Condition": { "ForAnyValue:StringLike" : {
  "aws:ResourceOrgPaths":["o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/*"]
}}
```

Kondisi berikut mengembalikan sumber daya `True` yang dimiliki oleh akun yang dilampirkan langsung ke ID `OUou-ab12-22222222`, tetapi bukan OU anak. Contoh berikut menggunakan operator [StringEquals](#) kondisi untuk menentukan persyaratan kecocokan yang tepat untuk ID OU dan bukan pencocokan 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 di-host 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, mengandalkan 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 menentukan 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 lebih lanjut.

`aws: ResourceOrg ID`

Gunakan kunci ini untuk membandingkan pengenalan AWS organisasi dalam Organizations yang menjadi sumber daya yang diminta dengan pengenalan 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`
 - Amazon Elastic Block Store - Semua tindakan
 - Amazon EC2
 - `ec2:AcceptTransitGatewayPeeringAttachment`
 - `ec2:AcceptVpcEndpointConnections`
 - `ec2:AcceptVpcPeeringConnection`
 - `ec2:CopyFpgaImage`
 - `ec2:CopyImage`
 - `ec2:CopySnapshot`
 - `ec2:CreateTransitGatewayPeeringAttachment`
 - `ec2:CreateVolume`
 - `ec2:CreateVpcEndpoint`
 - `ec2:CreateVpcPeeringConnection`
 - `ec2>DeleteTransitGatewayPeeringAttachment`
 - `ec2>DeleteVpcPeeringConnection`
 - `ec2:RejectTransitGatewayPeeringAttachment`
 - `ec2:RejectVpcEndpointConnections`

- `ec2:RejectVpcPeeringConnection`
- Amazon EventBridge
 - `events:PutEvents`— EventBridge `PutEvents` memanggil 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 peristiwa dari AWS akun lain](#) di Panduan EventBridge Pengguna Amazon.
- Amazon GuardDuty
 - `guardduty:AcceptAdministratorInvitation`
- Amazon Macie
 - `macie2:AcceptInvitation`
- Amazon Route 53
 - `route53:AssociateVpcWithHostedZone`
 - `route53:CreateVPCAssociationAuthorization`
 - `route53>DeleteVPCAssociationAuthorization`
 - `route53:DisassociateVPCFromHostedZone`
 - `route53:ListHostedZonesByVPC`
- AWS Security Hub
 - `securityhub:AcceptAdministratorInvitation`
- Amazon WorkSpaces
 - `workspaces:DescribeWorkspaceImages`
- Tipe data - [String](#)
- Jenis nilai - Bernilai tunggal

 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 prinsipal 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.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "DenyPutObjectToS3ResourcesOutsideMyOrganization",
    "Effect": "Deny",
    "Action": "s3:PutObject",
    "Resource": "arn:partition:s3:::policy-genius-dev/*",
    "Condition": {
      "StringNotEquals": {
        "aws:ResourceOrgID": "${aws:PrincipalOrgID}"
      }
    }
  }
}
```

Note

Beberapa Layanan AWS memerlukan akses ke sumber daya yang AWS dimiliki yang di-host 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, mengandalkan 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 menentukan pengecualian untuk sumber daya milik layanan. Anda dapat membuat kebijakan serupa untuk membatasi akses ke sumber daya dalam organisasi Anda 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 lebih lanjut.

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 terpercaya.](#)

`aws:ResourceTag/tag-kunci`

Gunakan kunci ini untuk membandingkan pasangan nilai kunci tag yang Anda tentukan dalam kebijakan dengan pasangan nilai kunci 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 sumber daya AWS](#).

- 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 - Bernilai tunggal

Kunci konteks ini diformat "`aws:ResourceTag/tag-key`": "`tag-value`" dengan `tag-key` dan `tag-value` adalah kunci tanda dan pasangan nilai. Kunci dan nilai 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.

Untuk contoh menggunakan `aws:ResourceTag` kunci untuk mengontrol akses ke sumber daya IAM, lihat [Mengontrol akses ke sumber daya AWS](#).

Untuk contoh menggunakan `aws:ResourceTag` kunci untuk mengontrol akses ke AWS sumber daya lain, lihat [Mengontrol akses ke AWS sumber daya menggunakan tag](#).

Untuk tutorial tentang menggunakan kunci kondisi `aws:ResourceTag` untuk kontrol akses berbasis atribut (ABAC), lihat [Tutorial IAM: Menentukan izin untuk mengakses sumber daya AWS berdasarkan tanda](#).

Properti permintaan

Gunakan tombol kondisi berikut untuk membandingkan detail tentang permintaan itu sendiri dan konten permintaan dengan properti permintaan yang Anda tentukan dalam kebijakan.

Daftar Isi

- [aws: CalledVia](#)
- [aws: CalledViaFirst](#)
- [aws: CalledViaLast](#)
- [AWS: melalui AWSService](#)
- [aws: CurrentTime](#)
- [aws: EpochTime](#)
- [aws: referer](#)
- [aws: RequestedRegion](#)
- [aws: RequestTag/tag-kunci](#)
- [aws: TagKeys](#)
- [aws: SecureTransport](#)
- [aws: SourceArn](#)
- [aws: SourceAccount](#)
- [aws: SourceOrgPaths](#)
- [aws: SourceOrg ID](#)
- [aws: UserAgent](#)

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 kredensi kepala sekolah untuk membuat permintaan berikutnya ke layanan lain. Kunci `aws:CalledVia` berisi daftar yang dipesan dari setiap layanan dalam rantai yang membuat permintaan atas nama utama.

Misalnya, Anda dapat menggunakan AWS CloudFormation untuk membaca dan menulis dari tabel Amazon DynamoDB. DynamoDB kemudian menggunakan enkripsi yang disediakan AWS Key Management Service oleh `aws:kms`.

- Ketersediaan – Kunci ini tersedia dalam permintaan saat layanan yang mendukung `aws:CalledVia` menggunakan kredensial prinsipal IAM untuk mengajukan permintaan ke layanan lain. Kunci ini tidak tersedia jika layanan menggunakan [peran layanan](#) atau [peran tertaut layanan](#) untuk melakukan panggilan atas prinsipal. Kunci ini juga tidak tersedia jika prinsipal menelepon langsung.
- Tipe data - [String](#) (daftar)
- Jenis nilai — Multivalued

Untuk menggunakan kunci `aws:CalledVia` kondisi dalam kebijakan, Anda harus memberikan prinsip layanan untuk mengizinkan atau menolak AWS permintaan layanan. AWS mendukung menggunakan prinsip layanan berikut dengan `aws:CalledVia`

Pemimpin layanan

`aoss.amazonaws.com`

`athena.amazonaws.com`

`backup.amazonaws.com`

`cloud9.amazonaws.com`

`cloudformation.amazonaws.com`

`databrew.amazonaws.com`

Pemimpin layanan

dataexchange.amazonaws.com

dynamodb.amazonaws.com

imagebuilder.amazonaws.com

kms.amazonaws.com

mgn.amazonaws.com

nimble.amazonaws.com

omics.amazonaws.com

ram.amazonaws.com

robomaker.amazonaws.com

servicecatalog-appregistry.amazonaws.com

sqlworkbench.amazonaws.com

ssm-guiconnect.amazonaws.com

Untuk memungkinkan atau menolak akses ketika setiap layanan membuat permintaan dengan menggunakan kredensial prinsipal, gunakan kunci kondisi [AWS: melalui AWSService](#). 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 AWS CloudFormation, yang memanggil DynamoDB, yang memanggil AWS KMS. Ini adalah tiga permintaan terpisah. Panggilan terakhir ke AWS KMS dilakukan oleh Pengguna 1 melalui AWS CloudFormation dan kemudian DynamoDB.

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 `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.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "KmsActionsIfCalledViaDynamodb",
      "Effect": "Allow",
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey",
        "kms:DescribeKey"
      ],
      "Resource": "arn:aws:kms:region: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 diberi nama `my-example-key` AWS KMS. AWS KMS Operasi ini hanya diperbolehkan jika beberapa permintaan disertakan dalam rantai. Permintaan pertama harus dilakukan melalui AWS CloudFormation dan yang terakhir melalui DynamoDB. Jika layanan lain membuat permintaan di tengah rantai, operasi masih diizinkan.

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

```

    "Effect": "Allow",
    "Action": [
      "kms:Encrypt",
      "kms:Decrypt",
      "kms:ReEncrypt*",
      "kms:GenerateDataKey",
      "kms:DescribeKey"
    ],
    "Resource": "arn:aws:kms:region: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 prinsipal IAM untuk menghubungi layanan lain. Mereka menunjukkan layanan pertama dan terakhir yang melakukan panggilan dalam rantai permintaan. Misalnya, asumsikan bahwa AWS CloudFormation memanggil layanan lain bernama `X Service`, yang memanggil DynamoDB, yang kemudian memanggil AWS KMS. Panggilan terakhir ke AWS KMS dilakukan oleh `User 1` via AWS CloudFormation, kemudian `X Service`, dan kemudian DynamoDB. Ini pertama kali dipanggil via AWS CloudFormation dan terakhir dipanggil melalui DynamoDB.

`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 tersedia jika layanan menggunakan [peran layanan](#) atau [peran tertaut layanan](#) untuk melakukan panggilan atas prinsipal. Kunci ini juga tidak tersedia jika prinsipal menelepon langsung.
- Tipe data - [String](#)
- Jenis nilai - Bernilai tunggal

aws: 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 tersedia jika layanan menggunakan [peran layanan](#) atau [peran tertaut layanan](#) untuk melakukan panggilan atas prinsipal. Kunci ini juga tidak tersedia jika prinsipal menelepon langsung.
- Tipe data - [String](#)
- Jenis nilai - Bernilai tunggal

AWS: melalui AWSService

Gunakan kunci ini untuk memeriksa apakah suatu AWS layanan membuat permintaan ke layanan lain atas nama Anda.

Kunci konteks permintaan memberikan `true` ketika layanan menggunakan kredensial dari prinsipal IAM untuk membuat permintaan atas nama prinsipal. Kunci konteks memberikan `false` jika layanan menggunakan [peran layanan](#) atau [peran tertaut layanan](#) untuk melakukan panggilan atas nama prinsipal. Kunci konteks permintaan juga memberikan `false` saat prinsipal langsung memanggil.

- Ketersediaan – Kunci ini selalu disertakan dalam konteks permintaan.
- Tipe data — [Boolean](#)
- Jenis nilai - Bernilai tunggal

Anda dapat menggunakan kunci kondisi ini untuk mengizinkan atau menolak akses berdasarkan apakah permintaan dibuat oleh layanan.

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 - Bernilai tunggal

aws: 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 - Bernilai tunggal

aws:referer

Gunakan kunci ini untuk membandingkan siapa yang mereferensikan permintaan di browser klien dengan 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 - Bernilai tunggal

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. `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.

aws: 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 AWS Wilayah untuk setiap layanan, lihat [Titik akhir dan kuota Layanan](#) di Referensi Umum Amazon Web Services

- Ketersediaan – Kunci ini selalu disertakan dalam konteks permintaan.
- Tipe data - [String](#)
- Jenis nilai - Bernilai tunggal

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 mengatasinya, lihat [NotAction dengan Deny](#).

ℹ 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 tombol `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 AWS Management Console 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).

```
{
  "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"
            ]
        }
    }
}

```

aws:RequestTag/tag-kunci

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 permintaan AWS](#).

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan saat pasangan nilai kunci 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 - Bernilai tunggal

Kunci konteks ini diformat "aws:RequestTag/*tag-key*": "*tag-value*" dengan *tag-key* dan *tag-value* adalah kunci tanda dan pasangan nilai. Kunci dan nilai tag tidak peka huruf besar/kecil. Ini berarti bahwa jika Anda menentukan "aws:RequestTag/TagKey1": "Value1" dalam elemen kondisi kebijakan Anda, maka kondisi akan cocok dengan kunci tag permintaan bernama salah satu TagKey1 atautagkey1, tetapi tidak keduanya.

Contoh ini menunjukkan bahwa meskipun kuncinya bernilai tunggal, Anda masih dapat menggunakan beberapa pasangan nilai kunci dalam permintaan jika kuncinya berbeda.

```
{
```

```
"Version": "2012-10-17",
"Statement": {
  "Effect": "Allow",
  "Action": "ec2:CreateTags",
  "Resource": "arn:aws:ec2:::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 saat Anda menggunakan kebijakan untuk mengontrol akses menggunakan tag, gunakan tombol `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 tag yang lewat dalam permintaan.
- Tipe data - [String](#) (daftar)
- Jenis nilai — Multivalued

Kunci konteks ini diformat `"aws:TagKeys": "tag-key"` dengan *key-tag* adalah daftar kunci tanda 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 [Kunci konteks multivaluasi](#).

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 ini yang menggunakan `aws:TagKeys`, lihat [Membuat Snapshot dengan Tag](#) dalam Panduan Pengguna Amazon EC2 untuk Instans Linux.

`aws:SecureTransport`

Gunakan kunci ini untuk memeriksa apakah permintaan dikirim menggunakan SSL. Konteks permintaan mengembalikan `true` atau `false`. Dalam kebijakan, Anda dapat mengizinkan tindakan tertentu hanya jika permintaan dikirim menggunakan SSL.

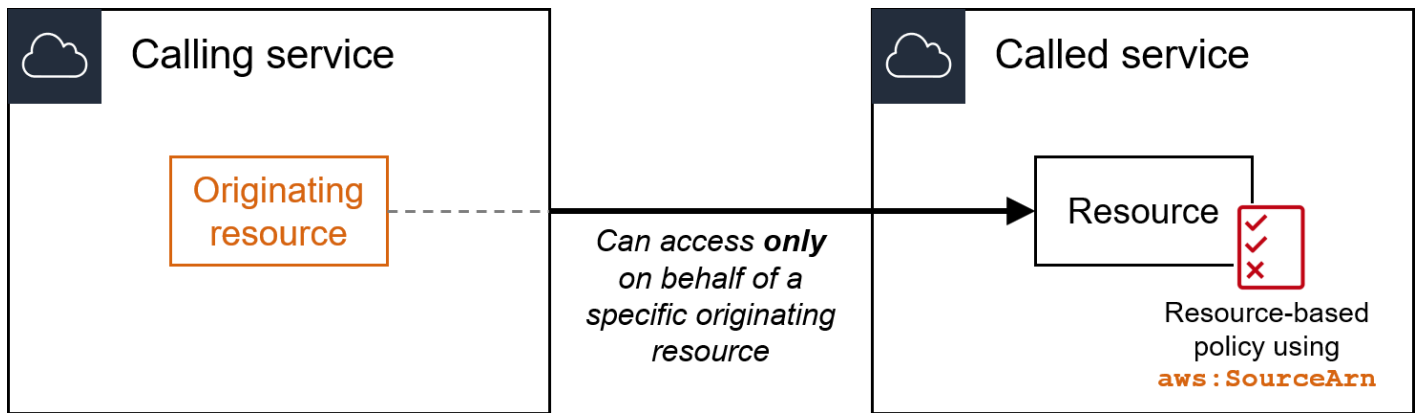
- Ketersediaan – Kunci ini selalu disertakan dalam konteks permintaan.
- Tipe data — [Boolean](#)
- Jenis nilai - Bernilai tunggal

`aws:SourceArn`

Gunakan kunci ini untuk membandingkan [Nama Sumber Daya Amazon \(ARN\)](#) sumber daya yang membuat service-to-service permintaan dengan ARN yang Anda tentukan dalam kebijakan, tetapi hanya jika permintaan dibuat oleh kepala layanan. AWS Ketika ARN sumber menyertakan ID akun, 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 langsung oleh [kepala AWS layanan](#) atas nama sumber daya yang konfigurasi memicu service-to-service permintaan. Layanan panggilan melewati ARN dari sumber daya asli ke layanan yang disebut.



Integrasi layanan berikut tidak mendukung kunci kondisi global ini:

Layanan panggilan (kepala layanan)	Disebut layanan (kebijakan berbasis sumber daya)	Deskripsi
logdelivery.elb.amazonaws.com	Bucket Amazon S3	Aktifkan log akses Elastic Load Balancing di bucket Amazon S3
logdelivery.elasticloadbalancing.amazonaws.com	Bucket Amazon S3	Aktifkan log akses Elastic Load Balancing di bucket Amazon S3

Note

Tidak semua integrasi layanan dengan AWS Security Token Service (AWS STS) dan AWS Key Management Service (AWS KMS) didukung. Lihat dokumentasi layanan panggilan untuk informasi lebih lanjut. Penggunaan `aws:SourceArn` dalam kebijakan kunci KMS untuk kunci yang digunakan oleh Layanan AWS melalui hibah kunci KMS dapat mengakibatkan perilaku yang tidak terduga.

- Tipe data - ARN, String

AWS merekomendasikan agar Anda menggunakan operator [ARN alih-alih operator string](#) saat membandingkan ARN.

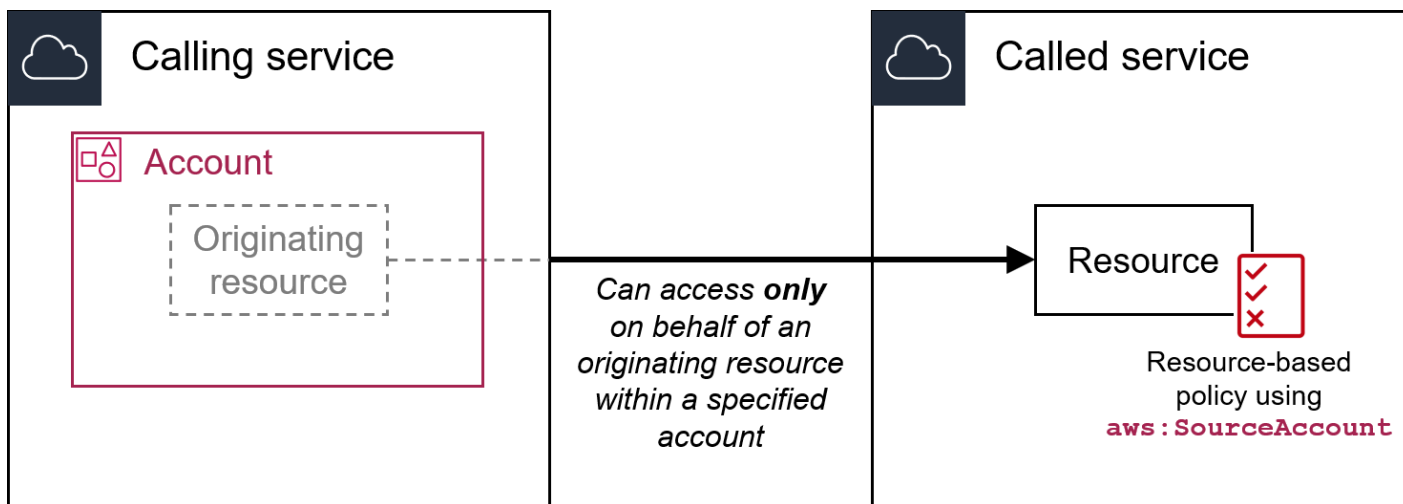
- Jenis nilai - Bernilai tunggal

Anda dapat menggunakan kunci kondisi ini untuk mencegah AWS layanan digunakan sebagai [wakil yang bingung](#) selama transaksi antar layanan. Gunakan kunci ini hanya dalam kebijakan berbasis sumber daya yang merupakan Principal prinsipal. Layanan AWS Tetapkan nilai kunci kondisi ini ke ARN sumber daya dalam permintaan. Misalnya, saat pembaruan bucket Amazon S3 memicu publikasi topik Amazon SNS, layanan Amazon S3 akan memanggil operasi API. `sns:Publish` Dalam kebijakan topik yang memungkinkan `sns:Publish` pengoperasian, tetapkan nilai kunci kondisi ke ARN bucket Amazon S3. Untuk informasi tentang bagaimana dan kapan kunci kondisi ini direkomendasikan, lihat dokumentasi untuk AWS layanan yang Anda gunakan.

`aws: SourceAccount`

Gunakan kunci ini untuk membandingkan ID akun sumber daya yang membuat service-to-service permintaan dengan ID akun yang Anda tentukan dalam kebijakan, tetapi hanya jika permintaan dibuat oleh kepala AWS layanan.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan hanya ketika panggilan ke sumber daya Anda dilakukan langsung oleh [kepala AWS layanan](#) atas nama sumber daya yang konfigurasi memicu service-to-service permintaan. Layanan panggilan meneruskan ID akun dari sumber daya asli ke layanan yang disebut.



Integrasi layanan berikut tidak mendukung kunci kondisi global ini:

Layanan panggilan (kepala layanan)	Disebut layanan (kebijakan berbasis sumber daya)	Deskripsi
logdelivery.elb.amazonaws.com	Bucket Amazon S3	Aktifkan log akses Elastic Load Balancing di bucket Amazon S3
logdelivery.elasticloadbalancing.amazonaws.com	Bucket Amazon S3	Aktifkan log akses Elastic Load Balancing di bucket Amazon S3

Note

Tidak semua integrasi layanan dengan AWS Security Token Service (AWS STS) dan AWS Key Management Service (AWS KMS) didukung. Lihat dokumentasi layanan panggilan untuk informasi lebih lanjut. Penggunaan `aws:SourceAccount` dalam kebijakan kunci KMS untuk kunci yang digunakan oleh Layanan AWS melalui hibah kunci KMS dapat mengakibatkan perilaku yang tidak terduga.

- Tipe data - [String](#)
- Jenis nilai - Bernilai tunggal

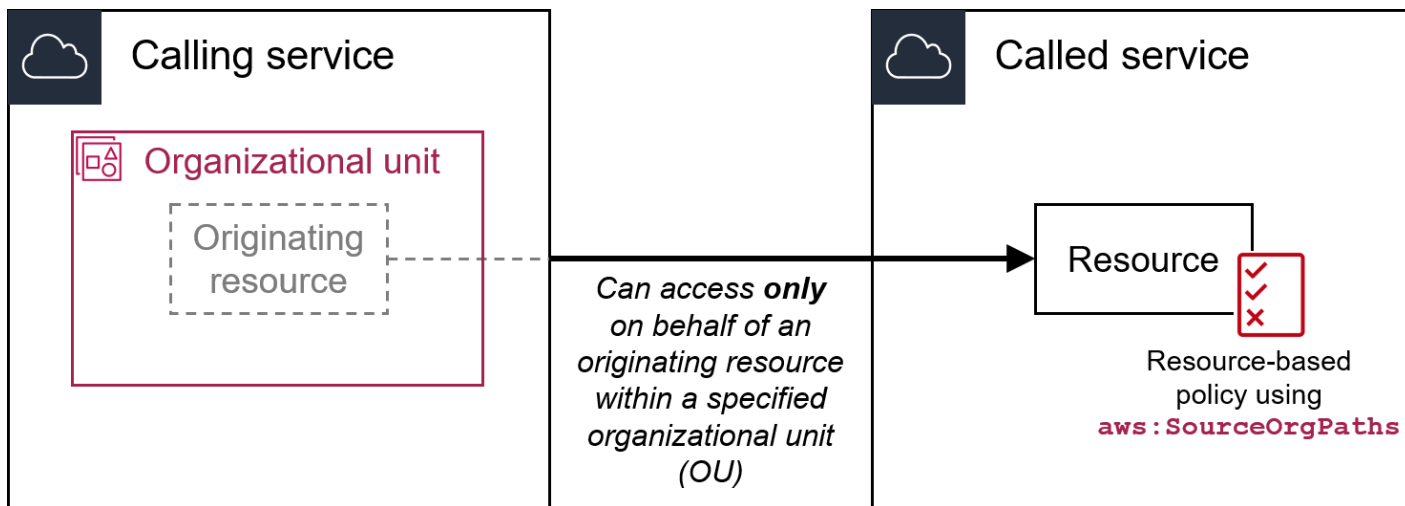
Anda dapat menggunakan kunci kondisi ini untuk mencegah AWS layanan digunakan sebagai [wakil yang bingung](#) selama transaksi antar layanan. Gunakan kunci ini hanya dalam kebijakan berbasis sumber daya yang merupakan `Principal` prinsipal. Layanan AWS Tetapkan nilai kunci kondisi ini ke ID akun sumber daya dalam permintaan. Misalnya, saat pembaruan bucket Amazon S3 memicu publikasi topik Amazon SNS, layanan Amazon S3 akan memanggil operasi API. `sns:Publish` Dalam kebijakan topik yang memungkinkan `sns:Publish` pengoperasian, tetapkan nilai kunci kondisi ke ID akun bucket Amazon S3. Untuk informasi tentang bagaimana dan kapan kunci kondisi ini direkomendasikan, lihat dokumentasi untuk AWS layanan yang Anda gunakan.

`aws:SourceOrgPaths`

Gunakan kunci ini untuk membandingkan AWS Organizations jalur sumber daya yang membuat service-to-service permintaan dengan jalur organisasi yang Anda tentukan dalam kebijakan, tetapi

hanya jika permintaan dibuat oleh kepala AWS layanan. Organizations path adalah representasi teks dari struktur entitas Organizations. 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 langsung oleh [kepala AWS layanan](#) atas nama sumber daya yang dimiliki oleh akun yang merupakan anggota organisasi. Layanan panggilan melewati jalur organisasi sumber daya asli ke layanan yang disebut.



Integrasi layanan berikut tidak mendukung kunci kondisi global ini:

Layanan panggilan (kepala layanan)	Disebut layanan (kebijakan berbasis sumber daya)	Deskripsi
logdelivery.elb.amazonaws.com	Bucket Amazon S3	Aktifkan log akses Elastic Load Balancing di bucket Amazon S3
logdelivery.elasticloadbalancing.amazonaws.com	Bucket Amazon S3	Aktifkan log akses Elastic Load Balancing di bucket Amazon S3
Semua kepala sekolah layanan	Bot Amazon Lex	Izinkan Layanan AWS untuk menggunakan bot Amazon Lex

Note

Tidak semua integrasi layanan dengan AWS Security Token Service (AWS STS) dan AWS Key Management Service (AWS KMS) didukung. Lihat dokumentasi layanan panggilan untuk informasi lebih lanjut. Penggunaan `aws:SourceOrgPaths` dalam kebijakan kunci KMS untuk kunci yang digunakan oleh Layanan AWS melalui hibah kunci KMS dapat mengakibatkan perilaku yang tidak terduga.

- Tipe data - [String](#) (daftar)
- Jenis nilai — Multivalued

Anda dapat menggunakan kunci kondisi ini untuk mencegah AWS layanan digunakan sebagai [wakil yang bingung](#) selama transaksi antar layanan. Gunakan kunci ini hanya dalam kebijakan berbasis sumber daya yang merupakan `Principal` prinsipal. Layanan AWS Tetapkan nilai kunci kondisi ini ke jalur organisasi sumber daya dalam permintaan. Misalnya, saat pembaruan bucket Amazon S3 memicu publikasi topik Amazon SNS, layanan Amazon S3 akan memanggil operasi API. `sns:Publish` Dalam kebijakan topik yang memungkinkan `sns:Publish` pengoperasian, tetapkan nilai kunci kondisi ke jalur organisasi bucket Amazon S3. Untuk informasi tentang bagaimana dan kapan kunci kondisi ini direkomendasikan, lihat dokumentasi untuk AWS layanan yang Anda gunakan.

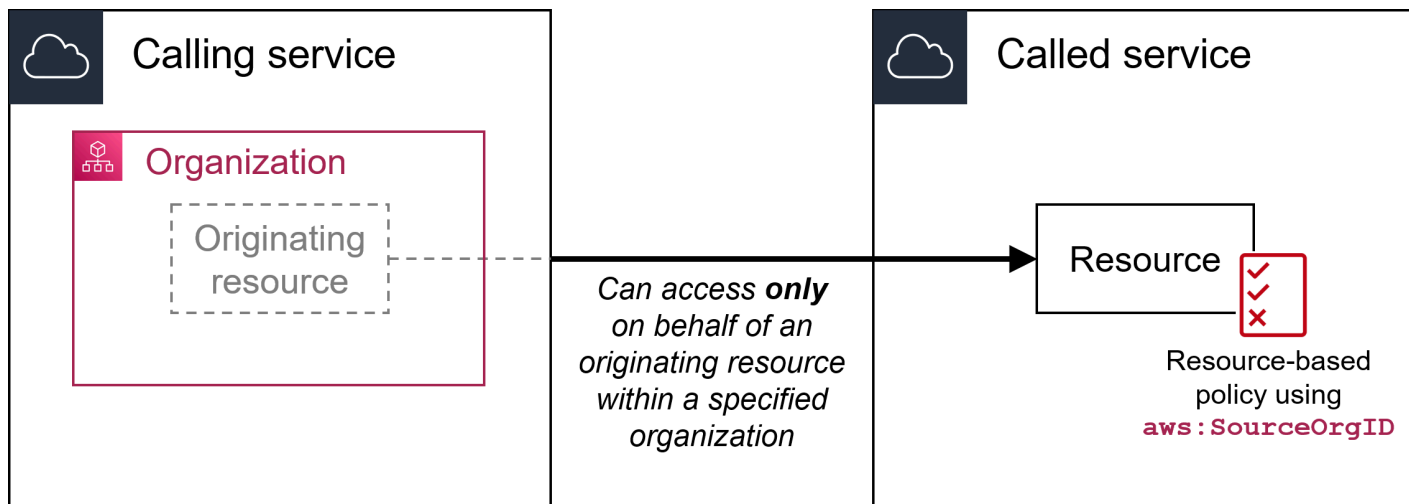
`aws:SourceOrgPaths` adalah kunci kondisi multivalued. Kunci multivalued dapat memiliki beberapa nilai dalam konteks permintaan. Anda harus menggunakan `ForAnyValue` atau `ForAllValues` mengatur operator dengan [operator kondisi string](#) untuk kunci ini. Untuk informasi selengkapnya tentang kunci kondisi multivalued ini, lihat [Kunci konteks multivaluasi](#).

`aws:SourceOrgID`

Gunakan kunci ini untuk membandingkan [ID organisasi](#) sumber daya yang membuat service-to-service permintaan dengan ID organisasi yang Anda tentukan dalam kebijakan, tetapi hanya jika permintaan dibuat oleh prinsipal AWS layanan. Saat Anda menambahkan dan menghapus akun ke organisasi 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 langsung oleh [kepala 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 disebut.



Integrasi layanan berikut tidak mendukung kunci kondisi global ini:

Layanan panggilan (kepala layanan)	Disebut layanan (kebijakan berbasis sumber daya)	Deskripsi
logdelivery.elb.amazonaws.com	Bucket Amazon S3	Aktifkan log akses Elastic Load Balancing di bucket Amazon S3
logdelivery.elasticloadbalancing.amazonaws.com	Bucket Amazon S3	Aktifkan log akses Elastic Load Balancing di bucket Amazon S3
Semua kepala sekolah layanan	Bot Amazon Lex	Izinkan Layanan AWS untuk menggunakan bot Amazon Lex

Note

Tidak semua integrasi layanan dengan AWS Security Token Service (AWS STS) dan AWS Key Management Service (AWS KMS) didukung. Lihat dokumentasi layanan panggilan untuk informasi lebih lanjut. Penggunaan `aws:SourceOrgID` dalam kebijakan kunci

KMS untuk kunci yang digunakan oleh Layanan AWS melalui hibah kunci KMS dapat mengakibatkan perilaku yang tidak terduga.

- Tipe data - [String](#)
- Jenis nilai - Bernilai tunggal

Anda dapat menggunakan kunci kondisi ini untuk mencegah AWS layanan digunakan sebagai [wakil yang bingung](#) selama transaksi antar layanan. Gunakan kunci ini hanya dalam kebijakan berbasis sumber daya yang merupakan Principal prinsipal. Layanan AWS Tetapkan nilai kunci kondisi ini ke ID organisasi sumber daya dalam permintaan. Misalnya, saat pembaruan bucket Amazon S3 memicu publikasi topik Amazon SNS, layanan Amazon S3 akan memanggil operasi API. `sns:Publish` Dalam kebijakan topik yang memungkinkan `sns:Publish` pengoperasian, tetapkan nilai kunci kondisi ke ID organisasi bucket Amazon S3. Untuk informasi tentang bagaimana dan kapan kunci kondisi ini direkomendasikan, lihat dokumentasi untuk AWS layanan yang Anda gunakan.

`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 - Bernilai tunggal

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.

Kunci kondisi lintas layanan lainnya

AWS STS [mendukung kunci kondisi federasi berbasis SAMP dan kunci kondisi lintas layanan untuk federasi OIDC](#). Kunci ini tersedia ketika pengguna yang difederasi menggunakan SAMP melakukan AWS operasi di layanan lain.

Kunci konteks IAM dan AWS STS 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: Condition](#).

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 konteks AWS federasi OIDC lintas layanan](#)
- [Kunci yang tersedia untuk federasi AWS STS berbasis SAML](#)
- [Kunci konteks federasi berbasis SAML AWS STS lintas layanan](#)
- [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:

saya: 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.

Gunakan kunci kondisi ini dalam kebijakan untuk mengizinkan entitas meneruskan peran, tetapi hanya jika peran tersebut terkait dengan sumber daya yang ditentukan. Anda dapat menggunakan wildcard (*) untuk memungkinkan operasi dilakukan pada tipe sumber daya yang spesifik tanpa membatasi Wilayah atau ID sumber daya. Misalnya, Anda dapat mengizinkan pengguna atau peran IAM untuk meneruskan peran apa pun ke layanan Amazon EC2 untuk digunakan dengan instans di Wilayah atau. us-east-1 us-west-1 Pengguna atau peran IAM tidak akan diizinkan untuk meneruskan peran ke layanan lain. Selain itu, Amazon EC2 tidak mengizinkan Amazon EC2 untuk menggunakan peran dengan instans di Wilayah lain.

```
{
  "Effect": "Allow",
  "Action": "iam:PassRole",
  "Resource": "*",
  "Condition": {
    "StringEquals": {"iam:PassedToService": "ec2.amazonaws.com"},
    "ArnLike": {
      "iam:AssociatedResourceARN": [
        "arn:aws:ec2:us-east-1:111122223333:instance/*",
        "arn:aws:ec2:us-west-1:111122223333:instance/*"
      ]
    }
  }
}
```

Note

AWS layanan yang mendukung [iam: PassedToService](#) juga mendukung kunci kondisi ini.

saya: AWSServiceName

Bekerja dengan [operator string](#).

Menentukan AWS layanan yang peran ini dilampirkan.

Dalam contoh ini, Anda mengizinkan entitas untuk membuat peran terkait layanan jika nama layanan adalah `access-analyzer.amazonaws.com`.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": "iam:CreateServiceLinkedRole",
    "Resource": "*",
    "Condition": {
      "StringLike": {
        "iam:AWSServiceName": "access-analyzer.amazonaws.com"
      }
    }
  }]
}
```

IAM:Sertifikasi FIDO

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.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": "iam:EnableMFADevice",
```

```

    "Resource": "*",
    "Condition": {
      "StringEquals": {
        "iam:RegisterSecurityKey" : "Create"
      }
    },
    {
      "Effect": "Allow",
      "Action": "iam:EnableMFADevice",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "iam:RegisterSecurityKey" : "Activate",
          "iam:FIDO-certification": "L1plus"
        }
      }
    }
  ]
}

```

IAM: Sertifikasi FIDO-FIPS-140-2-

Bekerja dengan [operator string](#).

Memeriksa perangkat MFA FIPS-140-2 tingkat sertifikasi validasi 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.

```

{
  "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: Sertifikasi FIDO-FIPS-140-3-

Bekerja dengan [operator string](#).

Memeriksa perangkat MFA FIPS-140-3 tingkat sertifikasi validasi 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.

```

{
  "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"
        }
      }
    }
  ]
}

```

saya: RegisterSecurityKey

Bekerja dengan [operator string](#).

Memeriksa status pemberdayaan 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.

```

{
  "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"
        }
      }
    }
  ]
}
```

saya: 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 yang terakhir diakses untuk kebijakan Organizations yang terakhir diakses untuk](#).

saya: 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 dipercaya 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, lihat [IAM: Teruskan peran IAM ke AWS layanan khusus](#).

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.

saya: `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](#).

`iam:ResourceTag/nama-kunci`

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 kustom ke sumber daya IAM dalam bentuk pasangan kunci-nilai. Untuk informasi selengkapnya tentang tag untuk sumber daya IAM, lihat [the section called “Menandai 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 *teks placeholder yang dicetak miring* dalam kebijakan contoh dengan informasi Anda sendiri. Lalu, ikuti petunjuk di [buat kebijakan](#) atau [ubah kebijakan](#).

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": "iam:DeleteUser",
    "Resource": "*",
    "Condition": {"StringEquals": {"iam:ResourceTag/status": "terminated"}}
  }]
}
```

Kunci yang tersedia untuk AWS federasi OIDC

Anda dapat menggunakan federasi OIDC untuk memberikan kredensial keamanan sementara kepada pengguna yang telah diautentikasi melalui OpenID Connect compliant OpenID Provider (OP) OpenID Connect (OIDC) ke penyedia identitas IAM OpenID Connect (OIDC) di akun Anda. AWS Contoh penyedia tersebut termasuk Login with Amazon, Amazon Cognito, Google, atau Facebook. Token identitas (`id_tokens`) dari OP OpenID Anda sendiri dapat digunakan, serta `id_token` yang dikeluarkan untuk akun layanan cluster Amazon Elastic Kubernetes Service. Dalam kasus tersebut, kunci kondisi tambahan tersedia ketika kredensial keamanan sementara digunakan untuk membuat permintaan. Anda dapat menggunakan kunci ini untuk menulis kebijakan yang membatasi akses pengguna gabungan 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 diikuti dengan klaim (:aud,, :azp:amr,:sub). Untuk peran yang digunakan oleh Amazon Cognito, kunci ditentukan menggunakan `cognito-identity.amazonaws.com` diikuti oleh klaim.

amr

Bekerja dengan [operator string](#).

Contoh: `cognito-identity.amazonaws.com:amr`

Jika Anda menggunakan Amazon Cognito untuk federasi OIDC, `cognito-identity.amazonaws.com:amr` kunci (Referensi Metode Otentikasi) mencakup informasi login tentang pengguna. Kunci ini memiliki banyak nilai, artinya jika Anda mengujinya dalam suatu kebijakan yang menggunakan [operator kumpulan kondisi](#). Kunci dapat berisi nilai-nilai berikut:

- Jika pengguna tidak autentikasi, kunci hanya berisi `unauthenticated`.
- Jika pengguna terautentikasi, kunci berisi nilai `authenticated` dan nama penyedia masuk yang digunakan dalam panggilan (`graph.facebook.com`, `accounts.google.com`, atau `www.amazon.com`).

Sebagai contoh, kondisi berikut dalam kebijakan kepercayaan untuk peran Amazon Cognito menguji apakah pengguna tidak terautentikasi:

```
"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](#).

Gunakan kunci kondisi `aud` untuk memverifikasi bahwa ID klien Google atau ID kolam identitas Amazon Cognito sesuai dengan yang Anda tentukan dalam kebijakan. Anda dapat menggunakan kunci `aud` dengan kunci `sub` untuk penyedia identitas yang sama.

Contoh:

- `graph.facebook.com:app_id`

- `accounts.google.com:aud`
- `cognito-identity.amazonaws.com:aud`

`graph.facebook.com:app_id` Bidang ini menyediakan konteks audiens yang cocok dengan bidang `aud` yang digunakan oleh penyedia identitas lainnya.

Kunci kondisi `accounts.google.com:aud` cocok dengan kolom Token ID Google berikut.

- `aud` untuk ID klien Google OAuth 2.0 dari aplikasi Anda, jika kolom `azp` belum ditetapkan. Ketika `azp` bidang diatur, `aud` bidang cocok dengan kunci [accounts.google.com:aud](#) 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.

Untuk informasi selengkapnya tentang kolom `aud` dan `azp` Google, lihat Panduan [Google Identity Platform OpenID Connect](#).

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:aud`.

```
{
  "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:aud": "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:oauth`. Nilai kolom azp sesuai dengan nilai kunci kondisi `accounts.google.com:aud`.

```

{
  "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:oauth": "aud-value",
          "accounts.google.com:sub": "sub-value"
        }
      }
    }
  ]
}

```

id

Bekerja dengan [operator string](#).

Contoh:

- `graph.facebook.com:id`
- `www.amazon.com:app_id`
- `www.amazon.com:user_id`

Gunakan kunci ini untuk memverifikasi bahwa aplikasi (atau situs) ID atau ID pengguna cocok dengan yang Anda tentukan dalam kebijakan. Cara ini berfungsi untuk Facebook atau Login with

Amazon. Anda dapat menggunakan kunci `app_id` dengan kunci `id` untuk penyedia identitas yang sama.

`aud`

Bekerja dengan [operator string](#).

Contoh: `accounts.google.com:aud`

Jika Anda menggunakan Google untuk federasi OIDC, kunci ini menentukan audiens Google (`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`
- `cognito-identity.amazonaws.com:sub`

Gunakan kunci ini untuk memverifikasi bahwa ID pengguna sesuai dengan ID yang Anda tentukan dalam kebijakan. Anda dapat menggunakan kunci `sub` dengan kunci `aud` untuk penyedia identitas yang sama.

```
{
  "Version": "2012-10-17",
  "Statement": [
    "Condition": {
      "StringEquals": {
        "oidc.eks.us-east-1.amazonaws.com/id/111122223333:aud":
"sts.amazonaws.com",
        "oidc.eks.us-east-1.amazonaws.com/id/111122223333:sub":
"system:serviceaccount:default:assumer"
      }
    }
  ]
}
```

Informasi lebih lanjut tentang federasi OIDC

Untuk informasi lebih lanjut tentang federasi OIDC, lihat berikut ini:

- [Panduan Pengguna Amazon Cognito](#).

- [Federasi OIDC](#)

Kunci konteks AWS federasi OIDC lintas layanan

Beberapa kunci kondisi federasi OIDC dapat digunakan dalam kebijakan kepercayaan peran untuk menentukan apa yang pengguna diizinkan untuk mengakses di layanan lain AWS. Ini adalah kunci kondisi berikut yang dapat digunakan dalam kebijakan kepercayaan peran ketika kepala sekolah federasi mengambil peran lain, dan dalam kebijakan sumber daya dari AWS layanan lain untuk mengotorisasi akses sumber daya oleh prinsipal federasi. Jika Anda menggunakan Amazon Cognito untuk federasi OIDC, maka kunci ini tersedia saat pengguna diautentikasi.

Pilih tombol kondisi untuk melihat deskripsi.

- [amr](#)
- [aud](#)
- [id](#)
- [sub](#)

Note

Tidak ada kunci kondisi federasi berbasis identitas web lain yang tersedia untuk digunakan setelah otentikasi dan otorisasi penyedia identitas eksternal (IDP) untuk operasi awal. AssumeRoleWithWebIdentity

Kunci yang tersedia untuk federasi AWS STS berbasis SAML

Jika Anda bekerja dengan [federasi berbasis SAMP](#) 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 `sam1 : 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 dengan `x []` dapat memiliki nilai yang merupakan daftar jenis yang ditentukan.

saml: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.

saml: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 SAMP tersedia untuk kebijakan IAM sebagai kuncinya. `saml:doc` Untuk informasi selengkapnya, lihat [Mengidentifikasi pengguna secara unik dalam federasi berbasis SAML](#).

`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 [Konfigurasi pernyataan SAMP untuk respons otentikasi](#).

`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"]
  }
}
```

Kunci konteks federasi berbasis SAML AWS STS lintas layanan

Beberapa kunci kondisi federasi berbasis SAMP dapat digunakan dalam permintaan berikutnya untuk mengotorisasi AWS operasi di layanan dan panggilan lain. AssumeRole Ini adalah kunci kondisi berikut yang dapat digunakan dalam kebijakan kepercayaan peran ketika kepala sekolah 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 penggunaan kunci ini, lihat [Tentang federasi berbasis SAMM 2.0](#).

Pilih tombol kondisi untuk melihat deskripsi.

- [saml:namequalifier](#)
- [saml:sub](#)
- [saml:sub_type](#)

Note

Tidak ada kunci kondisi federasi berbasis SAML lainnya 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 kepala AWS CodeArtifact layanan.

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 lebih lanjut tentang token pembawa, lihat [Menggunakan token pembawa](#).

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.

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 prinsipal saat mendapatkan token AWS STS pembawa.

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 lebih lanjut tentang token pembawa, lihat [Menggunakan token pembawa](#).

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. Kuncinya tidak tersedia untuk operasi `assume-role` AWS STS .

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 prinsipal memberikan ID eksternal sambil mengambil peran menggunakan AWS API AWS CLI atau.

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 [Cara menggunakan ID eksternal saat memberikan akses ke sumber daya AWS Anda kepada 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 nilai kunci 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 sambil mengasumsikan peran menggunakan operasi API. `AWS STS AssumeRole`

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 menyetel kunci konteks dalam token sesi yang dihasilkan.

Anda dapat menggunakan kunci ini dalam kebijakan kepercayaan peran untuk menerapkan kontrol akses berbutir halus berdasarkan pengguna atau atributnya saat mereka mengambil peran. Misalnya, Anda dapat mengonfigurasi Amazon Redshift sebagai aplikasi Pusat Identitas IAM untuk mengakses sumber daya Amazon S3 atas nama tenaga kerja atau identitas federasi Anda.

Kebijakan kepercayaan peran berikut memungkinkan prinsipal layanan Amazon Redshift untuk berperan dalam akun. `111122223333` Ini juga memberikan izin kepada prinsipal layanan Amazon Redshift untuk menyetel kunci konteks dalam permintaan, selama set nilai `identitystore:UserId` kunci-konteks. `1111-22-3333-44-5555` Setelah peran diasumsikan, aktivitas muncul di AWS CloudTrail log di dalam `AdditionalEventData` elemen, yang berisi pasangan nilai kunci konteks sesi yang ditetapkan oleh penyedia konteks dalam permintaan peran asumsikan. Ini memudahkan administrator membedakan di antara sesi peran saat peran digunakan oleh prinsipal yang berbeda. Pasangan kunci-nilai ditetapkan oleh penyedia konteks yang ditentukan, bukan oleh AWS CloudTrail atau. AWS STS Ini memberi penyedia konteks kontrol atas konteks apa yang disertakan dalam CloudTrail log dan informasi sesi.

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "redshift.amazonaws.com"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:SetContext"
      ],
      "Condition": {
        "ForAllValues:ArnEquals": {
          "sts:RequestContextProviders": [
            "arn:aws:iam::aws:contextProvider/IdentityCenter"
          ]
        },
        "StringEquals": {
          "aws:SourceAccount": "111122223333",
          "sts:RequestContext/identitystore:UserId":
"1111-22-3333-44-5555"
        }
      }
    }
  ]
}

```

sts: RequestContextProviders

Bekerja IT [Operator ARN](#).

Gunakan kunci ini untuk membandingkan ARN penyedia konteks 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 sambil mengasumsikan peran menggunakan operasi API. `AWS STS AssumeRole`

Contoh kondisi berikut memeriksa apakah penyedia konteks ARN yang diteruskan dalam permintaan cocok dengan ARN yang ditentukan dalam kondisi kebijakan kepercayaan peran.

```
"Condition": {
```

```
"ForAllValues:ArnEquals": {
  "sts:RequestContextProviders": [
    "arn:aws:iam::aws:contextProvider/IdentityCenter"
  ]
}
```

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 prinsipal mengambil peran menggunakan, perintah CLI peran apa pun AWS Management Console, 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 apa pun sebagai [variabel](#). Anda tidak dapat menggunakan kunci kondisi multivalued sebagai variabel.

```
{
```



```

"Version": "2012-10-17",
"Statement": [
  {
    "Sid": "RoleTrustPolicyRequireUsernameForSessionName",
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Principal": {"AWS": "arn:aws:iam::111122223333:root"},
    "Condition": {
      "StringLike": {"sts:RoleSessionName": "${aws:username}"}
    }
  }
]
}

```

Saat 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, kami menyarankan Anda menggunakan 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 prinsipal memberikan identitas sumber sambil mengasumsikan peran menggunakan perintah CLI AWS STS `peran` apa pun, atau operasi API. `AWS STS AssumeRole`

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 dikirimkan ke 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 di dalamnya 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 akses lebih lanjut ke AWS sumber daya berdasarkan nilai identitas sumber dalam permintaan berikutnya.

Kebijakan kepercayaan peran berikut memungkinkan pengguna IAM AdminUser untuk mengambil peran dalam akun111122223333. Ini juga memberikan izin ke AdminUser untuk mengatur identitas sumber, selama identitas sumber yang ditetapkan adalah DiegoRamirez.

```
{
  "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: 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 syarat untuk layanan AWS.

Setiap AWS layanan dapat menentukan tindakan, sumber daya, dan kunci konteks kondisi untuk digunakan di kebijakan IAM. Untuk daftar layanan AWS dan tindakan, sumber daya, dan kunci konteks syarat, lihat [Tindakan, sumber daya, dan kunci syarat](#) dalam 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 Akun AWS dan sumber daya Anda.

Topik

- [Identitas](#)
- [Kredensial \(RDS, access key, dan perangkat MFA\)](#)
- [Izin dan kebijakan](#)
- [Federasi dan delegasi](#)
- [IAM dan produk lainnya AWS](#)
- [Praktik kerahasiaan umum](#)
- [Sumber daya umum](#)

Identitas

Konsultasikan sumber daya ini untuk membuat, mengelola, dan menggunakan identitas.

- [Kelola identitas di Pusat Identitas IAM](#) - Informasi prosedural tentang membuat pengguna dan grup di Pusat Identitas IAM.
- [Identitas IAM \(pengguna, pengguna, dan peran\)](#)— Diskusi mendalam tentang pengguna, grup, 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 untuk pengguna IAM.

- [Mengelola kata sandi pengguna di AWS](#)— Menjelaskan opsi untuk mengelola kata sandi untuk pengguna IAM di akun Anda.
- [Mengelola kunci akses untuk pengguna IAM](#)— Menjelaskan cara kerja kunci akses dan bagaimana Anda dapat menggunakannya untuk melakukan panggilan terprogram. 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.

- [Menggunakan otentikasi multi-faktor \(MFA\) di AWS](#) – 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 [AWS Security Credentials](#) di panduan.Referensi Umum AWS

Izin dan kebijakan

Pelajari cara kerja kebijakan IAM dan temukan tip tentang cara terbaik untuk memberikan izin:

- [Kebijakan dan Izin di IAM](#) – 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.
- [Memvalidasi 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 Policy Generator](#) - Buat kebijakan khusus dengan memilih produk dan tindakan dari daftar.
- [IAM Policy Simulator](#) — Uji apakah suatu kebijakan akan mengizinkan atau menolak permintaan tertentu. AWS

Federasi dan delegasi

Anda dapat memberikan akses ke sumber daya Akun AWS untuk pengguna yang diautentikasi (masuk) di tempat lain. Ini dapat berupa pengguna IAM di pengguna lain Akun AWS (dikenal sebagai delegasi), pengguna yang diautentikasi dengan proses masuk organisasi Anda, atau pengguna dari penyedia identitas Internet seperti Login with Amazon, Facebook, Google, atau penyedia identitas kompatibel OpenID Connect (OIDC) lainnya. Dalam kasus ini, pengguna mendapatkan kredensial keamanan sementara untuk mengakses AWS sumber daya.

- [Tutorial IAM: Delegasikan akses AWS S3 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 ke dalam AWS setelah diautentikasi di luar. AWS

IAM dan produk lainnya AWS

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.
- [Menggunakan profil instans](#)— Menjelaskan cara menggunakan peran IAM untuk menyediakan kredensial secara 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 Folder Khusus Pengguna di Amazon S3 Bucket](#) — Membahas cara membiarkan 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.

- [Pelatihan Dasar tentang Izin Tingkat Sumber Daya RDS](#) – 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 masing-masing item database DynamoDB atau atribut (atau keduanya).

[Memulai dengan Kontrol Akses Berbutir Halus untuk DynamoDB](#)

Praktik kerahasiaan umum

Temukan kiat dan panduan ahli tentang cara terbaik untuk mengamankan sumber daya Akun AWS dan Anda:

- [Praktik Terbaik untuk Keamanan, Identitas, & Kepatuhan](#) — Temukan sumber daya untuk mengelola keamanan Akun AWS dan produk, termasuk saran untuk arsitektur keamanan, penggunaan IAM, enkripsi dan keamanan data, dan banyak lagi.
- [Identity and Access Management](#) — The AWS Well-Architected Framework membantu Anda memahami konsep kunci, prinsip desain, dan praktik terbaik arsitektur untuk merancang dan menjalankan beban kerja di cloud.
- [Praktik terbaik keamanan di IAM](#)— Menawarkan rekomendasi untuk cara menggunakan IAM untuk membantu mengamankan sumber daya Akun AWS dan Anda.
- [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 mendiskusikan pertanyaan teknis terkait IAM dengan AWS masyarakat.
- [Kelas & Lokakarya](#) - Tautan ke kursus berbasis peran dan khusus, selain laboratorium mandiri untuk membantu mempertajam keterampilan Anda AWS 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 aplikasi. AWS
- [Memulai Pusat Sumber Daya](#) — Pelajari cara menyiapkan Akun AWS, bergabung dengan AWS komunitas, dan meluncurkan aplikasi pertama Anda.
- [Hands-On Tutorial](#) - Ikuti step-by-step tutorial untuk meluncurkan aplikasi pertama Anda. AWS
- [AWS Whitepaper](#) — Tautan ke daftar lengkap AWS whitepaper teknis, yang mencakup topik-topik seperti arsitektur, keamanan, dan ekonomi dan ditulis oleh AWS Solutions Architects atau pakar teknis lainnya.
- [AWS Support Pusat](#) — Hub untuk membuat dan mengelola AWS Support kasus Anda. Juga termasuk tautan ke sumber daya bermanfaat lainnya, seperti forum, FAQ teknis, status kesehatan layanan, dan. AWS Trusted Advisor
- [AWS Support](#)— Halaman web utama untuk informasi tentang AWS Support, saluran dukungan respons cepat untuk membantu Anda membangun dan menjalankan aplikasi di cloud. one-on-one
- [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, lisensi, dan akses situs Anda; 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 terprogram menggunakan API Kueri. Permintaan API Kueri adalah permintaan HTTPS yang harus berisi parameter `Action` untuk menunjukkan tindakan yang akan dilakukan. IAM dan AWS STS dukungan 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 masing-masing tindakan di [Referensi API IAM atau Referensi AWS Security Token Service API](#).

Tip

Alih-alih membuat panggilan langsung ke operasi IAM atau AWS STS API, Anda dapat menggunakan salah satu AWS SDK. SDK AWS terdiri atas pustaka dan kode sampel untuk berbagai bahasa dan platform pemrograman (Java, Ruby, .NET, iOS, Android, dan banyak lagi). AWS menyediakan cara yang nyaman untuk membuat akses terprogram ke IAM dan . Misalnya, SDK menangani tugas seperti menandatangani permintaan secara kriptografis (lihat di bawah ini), mengelola kesalahan, dan mencoba kembali permintaan secara otomatis. Untuk informasi tentang SDK AWS, termasuk cara mengunduh dan menginstalnya, lihat halaman [Alat untuk Amazon Web Services](#).

Untuk detail tentang tindakan API dan kesalahan, lihat [Referensi API IAM atau Referensi AWS Security Token Service API](#).

Titik akhir

IAM dan AWS STS masing-masing memiliki satu titik akhir global:

- (IAM) <https://iam.amazonaws.com>
- (AWS STS) <https://sts.amazonaws.com>

Note

AWS STS juga mendukung pengiriman permintaan ke titik akhir regional di samping titik akhir global. Sebelum Anda dapat menggunakan AWS STS di suatu Wilayah, Anda harus terlebih dahulu mengaktifkan STS di Wilayah tersebut untuk Akun AWS. Untuk informasi selengkapnya tentang mengaktifkan Wilayah tambahan untuk AWS STS, lihat [Mengelola AWS STS dalam Wilayah AWS](#).

Untuk informasi selengkapnya tentang titik akhir dan Wilayah AWS untuk semua layanan, lihat [Titik akhir dan kuota layanan](#) 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 kredensial Pengguna root akun AWS untuk pekerjaan sehari-hari dengan IAM. Anda dapat menggunakan kredensial untuk pengguna IAM atau Anda dapat menggunakan AWS STS untuk membuat kredensial keamanan sementara.

Untuk menandatangani permintaan API Anda, sebaiknya gunakan AWS Signature Versi 4. Untuk informasi tentang menggunakan Signature Versi 4, buka [Proses Penandatanganan Signature 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 hal berikut:

- [Kredensial Keamanan AWS](#) Menyediakan informasi umum tentang tipe kredensial yang digunakan untuk mengakses AWS.
- [Praktik terbaik keamanan di IAM](#). Menyajikan daftar saran penggunaan layanan IAM untuk membantu mengamankan sumber daya AWS 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
AccessAnalyzerServiceRolePolicy — Izin Ditambahkan	IAM Access Analyzer menambahkan dukungan untuk izin untuk mengambil status saat ini dari blokir akses publik untuk snapshot Amazon EC2 ke izin tingkat layanan. AccessAnalyzerServiceRolePolicy	23 Januari 2024
AccessAnalyzerServiceRolePolicy — Izin Ditambahkan	IAM Access Analyzer menambahkan aliran dan tabel DynamoDB ke izin tingkat layanan. AccessAnalyzerServiceRolePolicy	Januari 11, 2024
AccessAnalyzerServiceRolePolicy — Izin Ditambahkan	IAM Access Analyzer menambahkan bucket direktori Amazon S3 ke izin tingkat layanan. AccessAnalyzerServiceRolePolicy	1 Desember 2023
IAMAccessAnalyzerReadOnlyAccess — Izin Ditambahkan	IAM Access Analyzer menambahkan izin ke IAM AccessAnalyzerReadOnlyAccess untuk memungkinkan Anda memeriksa apakah pembaruan kebijakan Anda memberikan akses tambahan. Izin ini diperlukan oleh IAM Access Analyzer untuk	26 November 2023

melakukan pemeriksaan kebijakan pada kebijakan Anda.

[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 khusus](#)

IAM Access Analyzer sekarang menyediakan pemeriksaan kebijakan khusus untuk memvalidasi bahwa kebijakan IAM mematuhi standar keamanan Anda sebelum penerapan.

26 November 2023

[AccessAnalyzerServiceRolePolicy — Izin Ditambahkan](#)

IAM Access Analyzer menambahkan tindakan IAM ke izin tingkat layanan untuk mendukung tindakan berikut: [AccessAnalyzerServiceRolePolicy](#)

26 November 2023

- Daftar entitas untuk kebijakan
- Menghasilkan detail layanan yang terakhir diakses
- Daftar informasi kunci akses

[Tindakan terakhir mengakses informasi dan dukungan pembuatan kebijakan untuk lebih dari 60 layanan dan tindakan tambahan](#)

IAM sekarang mendukung tindakan informasi yang diakses terakhir dan [menghasilkan kebijakan dengan informasi tingkat tindakan](#) untuk lebih dari 60 layanan tambahan, bersama dengan daftar tindakan yang informasi terakhir diakses tindakan yang tersedia.

1 November 2023

[Tindakan terakhir mengakses dukungan informasi untuk lebih dari 140 layanan](#)

IAM sekarang menyediakan informasi tindakan yang diakses terakhir untuk lebih dari 140 layanan, bersama dengan daftar tindakan yang informasi terakhir diakses tindakan yang 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 tipe sumber daya baru](#)

IAM Access Analyzer menambahkan dukungan untuk jenis sumber daya berikut:

25 Oktober 2022

- Cuplikan volume Amazon EBS
- Repositori Amazon ECR
- Sistem file Amazon EFS
- Cuplikan Amazon RDS DB
- Cuplikan cluster Amazon RDS DB
- Topik Amazon SNS

[U2F penghentian dan pemutakhiran /FIDO WebAuthn](#)

Menghapus menyebutkan U2F sebagai opsi MFA dan menambahkan informasi tentang, FIDO2 WebAuthn, dan kunci keamanan FIDO.

31 Mei 2022

[Pembaruan ketahanan di IAM](#)

Menambahkan informasi tentang mempertahankan akses ke kredensial IAM ketika suatu peristiwa mengganggu komunikasi antara. Wilayah AWS

Mei 16, 2022

[Kunci kondisi global baru untuk sumber daya](#)

Sekarang Anda dapat mengontrol akses ke sumber daya berdasarkan akun, Unit Organisasi (OU), atau organisasi AWS Organizations yang berisi sumber daya Anda. Anda dapat menggunakan kunci kondisi `aws:ResourceAccount`, `aws:ResourceOrgID`, dan `aws:ResourceOrgPaths` global dalam kebijakan IAM.

27 April 2022

[Contoh kode untuk IAM menggunakan AWS SDK](#)

Contoh kode yang ditambahkan yang menunjukkan cara menggunakan IAM dengan kit pengembangan AWS perangkat lunak (SDK). Contoh dibagi menjadi kutipan kode yang menunjukkan 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 diagram alir logika evaluasi kebijakan](#)

Pembaruan pada diagram alir logika evaluasi kebijakan dan teks terkait dalam [Menentukan apakah permintaan diizinkan atau ditolak dalam bagian akun](#).

17 November 2021

Pembaruan praktik terbaik keamanan	Menambahkan informasi tentang membuat pengguna administratif alih-alih menggunakan kredensial pengguna root, menghapus praktik terbaik menggunakan grup pengguna 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 pokok yang berbeda di akun yang sama.	5 Oktober 2021
Pembaruan untuk kunci kondisi bernilai tunggal dan multivaluasi	Perbedaan antara kunci kondisi bernilai tunggal dan multivalued sekarang dijelaskan secara lebih rinci. Jenis nilai ditambahkan ke setiap kunci konteks kondisi AWS global .	30 September 2021
IAM Access Analyzer mendukung Titik Akses Multi-Wilayah Amazon S3	IAM Access Analyzer mengidentifikasi bucket Amazon S3 yang memungkinkan akses publik dan lintas akun, termasuk yang menggunakan Titik Akses Multi-Wilayah Amazon S3.	2 September 2021
AWS pembaruan kebijakan terkelola - Pembaruan ke kebijakan yang ada	IAM Access Analyzer memperbarui kebijakan AWS terkelola yang 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 jejak lintas akun](#)

Sekarang Anda dapat menggunakan IAM Access Analyzer untuk membuat 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](#)

29 Juni 2021

IAM Access Analyzer memperpanjang validasi kebijakan dengan menambahkan pemeriksaan kebijakan baru yang memvalidasi kondisi yang termasuk 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 berpasangan tidak ada](#)
- [Saran - Izinkan NotAction dengan kunci kondisi tag](#)

[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 tindakan informasi yang terakhir diakses di konsol IAM tentang terakhir kali prinsipal IAM menggunakan tindakan untuk layanan berikut: Amazon EC2, IAM, Lambda, dan tindakan pengelolaan 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 mengendalikan tindakan yang diambil dengan peran yang diasumsikan](#)

Administrator dapat mengonfigurasi peran IAM untuk mengharuskan identitas melewati 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	Sekarang Anda dapat menggunakan IAM Access Analyzer 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 sekarang 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 AndaAkun 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. Sekarang Anda dapat menemukan daftar layanan AWS dan tindakan, sumber daya, serta kunci konteks kondisinya di Referensi Otorisasi Layanan.

16 November 2020

[Pengguna IAM durasi sesi peran lebih lama](#)

Pengguna IAM sekarang dapat memiliki durasi sesi peran yang lebih lama saat beralih peran di AWS Management Console, 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 Service Quotas 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 akan dikirimkan ke AWS Support.

25 Juni 2020

[Informasi terakhir yang diakses di IAM sekarang mencakup tindakan manajemen Amazon S3](#)

3 Juni 2020

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 CLI atau API AWS 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.

[Penambahan Bab Keamanan](#)

Bab keamanan membantu Anda memahami 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.

29 April 2020

[sts: RoleSessionName](#)

Sekarang Anda dapat menyusun kebijakan yang memberikan izin berdasarkan nama sesi yang ditentukan oleh penanggung jawab saat mengambil peran.

21 April 2020

[AWS pembaruan halaman masuk](#)

Saat masuk di halaman AWS masuk utama, Anda tidak dapat memilih untuk masuk sebagai pengguna IAM Pengguna root akun AWS atau pengguna IAM. Ketika Anda melakukannya, label pada halaman menunjukkan apakah Anda harus memberikan alamat email pengguna root Anda atau informasi pengguna IAM Anda. Dokumentasi ini mencakup tangkapan layar yang diperbarui untuk membantu Anda memahami halaman masuk AWS.

4 Maret 2020

[AWS: via AWSService dan
aws: tumbol 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

[Evaluasi kebijakan lintas akun](#)

Kini 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 operasi `AssumeRoleWithSAML` atau `AssumeRoleWithWebIdentity` , Anda dapat meneruskan atribut dari identitas perusahaan Anda ke AWS.

22 November 2019

[Kontrol akses untuk grup Akun AWS di AWS Organizations](#)

Anda sekarang dapat merferensikan unit organisasi (OU) dari AWS Organizations dalam kebijakan IAM. Jika Anda menggunakan Organizations untuk mengatur akun Anda ke dalam OU, Anda dapat meminta prinsipal tersebut milik OU tertentu sebelum memberikan akses ke sumber daya Anda. Prinsipal 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 AWS Management Console, AWS CLI, dan API AWS 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 di AWS](#)

Pelajari bagaimana kontrol akses berbasis atribut (ABAC) bekerja di AWS menggunakan tanda, dan bagaimana ia membandingkannya dengan model otorisasi AWS biasa. 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 sumber daya AWS yang diperlukan untuk pekerjaan mereka saja.

3 Oktober 2019

[AWS STS GetAccessKeyInfo operasi](#)

Anda dapat meninjau access key AWS di kode Anda untuk menentukan apakah kunci tersebut berasal dari akun yang Anda miliki. Anda dapat melewati access key ID menggunakan perintah [aws sts get-access-key-info](#) AWS CLI atau Operasi API [GetAccessKeyInfo](#) AWS.

24 Juli 2019

[Layanan Viewing Organizations terakhir mengakses informasi di IAM](#)

Anda sekarang dapat melihat layanan informasi yang terakhir diakses untuk AWS Organizations entitas atau kebijakan di AWS Organizations bagian konsol IAM. Anda juga dapat menggunakan AWS CLI atau API AWS untuk mengambil laporan data. Data ini mencakup informasi tentang layanan yang diizinkan yang terakhir kali dicoba diakses dan kapan oleh penanggung jawab di akun Organizations. Anda dapat menggunakan informasi ini untuk mengidentifikasi izin yang tidak perlu, sehingga Anda dapat memperbaiki kebijakan Organizations Anda agar dapat mematuhi prinsip hak istimewa sekecil mungkin.

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 STSKompatibilitas 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 valid di Wilayah AWS 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 wilayah AWS](#)

Sekarang Anda dapat membuat kebijakan yang memungkinkan administrator untuk mengaktifkan dan menonaktifkan Wilayah Asia Pacific (Hong Kong) (ap-east-1).

24 April 2019

[Pengguna IAM halaman kredensial keamanan saya](#)

Pengguna IAM sekarang dapat mengelola semua kredensial mereka sendiri di laman Kredensial Keamanan Saya. Halaman AWS Management Console 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.

24 Januari 2019

[Akses API penasihat](#)

Sekarang Anda dapat menggunakan AWS CLI dan API AWS untuk melihat informasi yang terakhir diakses layanan.

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 AWS Management Console.	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 prinsip-prinsip IAM.	17 Mei 2018
aws: RequestedRegion	Kunci kondisi baru menyediakan cara yang lebih mudah untuk menggunakan kebijakan IAM untuk mengontrol akses ke AWS Wilayah.	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	Pengalaman AWS masuk yang diperbarui memungkinkan pengguna root dan pengguna IAM menggunakan tautan Masuk ke Konsol di halaman beranda. AWS Management Console	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 Auto Scaling	Izin tingkat sumber daya dapat mengontrol akses ke dan izin untuk sumber daya Auto Scaling.	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

[Peran terkait layanan](#)

Peran terkait layanan memberikan cara yang lebih mudah dan aman untuk mendelegasikan izin ke layanan AWS.

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.