



Panduan Developer

# Amazon Cognito



# Amazon Cognito: Panduan Developer

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 antara para pelanggan, atau dengan cara apa pun yang menghina atau mendiskreditkan Amazon. Semua merek dagang lain yang tidak dimiliki oleh Amazon merupakan milik dari pemiliknya masing-masing, yang mungkin berafiliasi dengan, terhubung ke, atau disponsori oleh Amazon.

---

# Table of Contents

Apa itu Amazon Cognito? .....	1
Kolam pengguna .....	2
Kolam identitas .....	3
Fitur Amazon Cognito .....	4
Kolam pengguna .....	4
Kolam identitas .....	7
Perbandingan kumpulan pengguna Amazon Cognito dan kumpulan identitas .....	9
Memulai dengan Amazon Cognito .....	13
Ketersediaan wilayah .....	13
Harga untuk Amazon Cognito .....	14
Cara kerja otentikasi .....	14
Otentikasi SDK .....	14
Otentikasi UI yang di-host .....	18
Otentikasi penyedia identitas pihak ketiga .....	21
Autentikasi kumpulan identitas .....	24
Istilah Amazon Cognito .....	27
Umum .....	28
Kolam pengguna .....	30
Kolam identitas .....	32
Bekerja dengan AWS SDK .....	33
Memulai dengan AWS .....	34
Mendaftar untuk Akun AWS .....	34
Buat pengguna dengan akses administratif .....	35
Memulai dengan kumpulan pengguna .....	37
Contoh React SPA .....	37
Membuat aplikasi .....	42
Buat lingkungan pengembang Lightsail .....	43
Contoh aplikasi seluler Flutter .....	44
Membuat aplikasi .....	49
Langkah selanjutnya .....	50
Buat kumpulan pengguna .....	51
Tambahkan klien aplikasi UI yang dihosting .....	55
Tambahkan penyedia sosial .....	59
Tambahkan penyedia SAMP .....	66

Memulai dengan kumpulan identitas .....	70
Buat kumpulan identitas di Amazon Cognito .....	70
Siapkan SDK .....	72
Integrasikan penyedia identitas .....	72
Dapatkan kredensial .....	73
Opsi memulai tambahan .....	74
Integrasi dengan aplikasi .....	76
Otentikasi dengan AWS Amplify .....	77
Membuat antarmuka pengguna (UI) dengan Amplify .....	78
Otentikasi dengan AWS SDK .....	79
Otorisasi dengan Izin Terverifikasi Amazon .....	79
Otorisasi API dengan Izin Terverifikasi .....	81
Contoh kebijakan untuk pengguna Amazon Cognito .....	84
Contoh kode .....	87
Identitas Amazon Cognito .....	89
Tindakan .....	89
Contoh lintas layanan .....	111
Penyedia Identitas Amazon Cognito .....	113
Tindakan .....	121
Skenario .....	238
Amazon Cognito Sync .....	363
Tindakan .....	363
Praktik terbaik aplikasi multi-penyewa .....	366
Pool pengguna per penyewa .....	368
Klien aplikasi per penyewa .....	370
Grup kumpulan pengguna per penyewa .....	372
Atribut kustom per penyewa .....	374
Rekomendasi keamanan multi-penyewa .....	376
Skenario Amazon Cognito yang umum .....	378
Otentikasi dengan kumpulan pengguna .....	378
Mengakses sumber daya sisi server Anda .....	379
Akses sumber daya dengan API Gateway dan Lambda .....	380
Akses AWS layanan dengan kumpulan pengguna dan kolam identitas .....	381
Mengautentikasi dengan pihak ketiga dan mengakses AWS layanan dengan kumpulan identitas .....	382
Akses AWS AppSync sumber daya dengan Amazon Cognito .....	383



Kolam pengguna Amazon Cognito .....	385
Fitur .....	386
Daftar .....	386
Masuk .....	387
UI yang Dihosting .....	388
Keamanan .....	388
Pengalaman pengguna khusus .....	389
Pemantauan dan analitik .....	389
Integrasi kumpulan identitas Amazon Cognito .....	390
Autentikasi .....	390
Alur otentikasi kumpulan pengguna .....	393
Klien aplikasi .....	403
Bekerja dengan perangkat .....	414
Menggunakan API dan titik akhir .....	420
Autentikasi API kumpulan pengguna .....	423
Memperbarui kumpulan pengguna .....	431
Konfigurasi SMS .....	433
Memperbarui kumpulan pengguna dengan AWS SDK, AWS CDK, atau REST API .....	433
Server UI dan OAuth yang dihosting .....	435
Menyiapkan UI yang dihosting dengan AWS Amplify .....	436
Menyiapkan UI yang dihosting dengan konsol Amazon Cognito .....	437
Melihat halaman masuk Anda .....	439
Hal-hal yang perlu diketahui tentang UI yang dihosting kumpulan pengguna Amazon Cognito .....	441
Mengkonfigurasi domain .....	442
Menyesuaikan halaman web bawaan .....	452
Cara menggunakan UI yang dihosting .....	459
Lingkup dan server sumber daya .....	478
Machine-to-machine Otorisasi M (M2M) .....	479
Tentang cakupan .....	480
Tentang server sumber daya .....	481
Menambahkan login melalui pihak ketiga .....	486
Cara kerja masuk federasi di kumpulan pengguna Amazon Cognito .....	486
Tanggung jawab aplikasi sebagai penyedia layanan dengan Amazon Cognito .....	488
Hal-hal yang perlu diketahui tentang kumpulan pengguna Amazon Cognito login pihak ketiga .....	488

Penyedia Identitas .....	489
Penyedia identitas sosial .....	495
Penyedia SAMP .....	503
Penyedia OIDC .....	536
Menentukan pemetaan atribut .....	546
Menautkan pengguna gabungan ke profil pengguna yang ada .....	551
Menggunakan pemacu Lambda .....	555
Pertimbangan penting .....	557
Menambahkan pemacu kumpulan pengguna .....	560
Acara pemacu Lambda kumpulan pengguna .....	561
Kumpulan pengguna Lambda memicu parameter umum .....	561
Lambda memicu sumber berdasarkan peristiwa .....	562
Lambda memicu sumber berdasarkan fungsi .....	569
Pemacu Lambda pra pendaftaran .....	572
Posting konfirmasi pemacu Lambda .....	581
Pemacu Lambda pra otentikasi .....	586
Posting otentikasi pemacu Lambda .....	590
Pemacu Lambda tantangan .....	595
Pemacu Lambda generasi pra token .....	610
Memigrasi pengguna pemacu Lambda .....	629
Pesan khusus Lambda pemacu .....	635
Pemacu Lambda pengirim kustom .....	642
Menggunakan analitik Amazon Pinpoint .....	660
Temukan pemetaan Wilayah Amazon Cognito dan Amazon Pinpoint .....	661
Mengintegrasikan aplikasi Anda dengan Amazon Pinpoint .....	665
Analitik .....	666
Mengelola pengguna .....	667
Mengizinkan pengguna mendaftar .....	668
Mendaftar dan mengonfirmasi akun pengguna .....	671
Menciptakan pengguna sebagai administrator .....	697
Menambahkan grup ke kumpulan pengguna .....	703
Mengelola dan mencari pengguna .....	706
Memulihkan akun pengguna .....	710
Mengimpor pengguna ke kolam pengguna .....	711
Atribut .....	729
Persyaratan kata sandi .....	742

Pengaturan Email .....	744
Konfigurasi email default .....	745
Konfigurasi email Amazon SES .....	746
Mengkonfigurasi akun email .....	751
Pengaturan pesan SMS .....	758
Menyiapkan pesan SMS untuk pertama kalinya di kumpulan pengguna Amazon Cognito ....	759
Menggunakan token .....	766
Menggunakan token ID .....	768
Menggunakan token akses .....	773
Menggunakan token penyegaran .....	776
Mencabut token .....	779
Memverifikasi JSON Web Token .....	780
Token cache .....	786
Mengakses sumber daya setelah masuk .....	789
Mengakses sumber daya dengan Izin Terverifikasi .....	379
Mengakses sumber daya dengan API Gateway dan AWS AppSync .....	792
Mengakses AWS sumber daya menggunakan kumpulan identitas .....	794
Menggunakan fitur keamanan .....	799
Menambahkan MFA .....	800
Menambahkan keamanan tingkat lanjut .....	812
AWS WAF ACL Web .....	829
Sensitivitas kasus .....	833
Perlindungan penghapusan .....	835
Mengelola pengungkapan pengguna .....	837
Kumpulan identitas Amazon Cognito .....	844
Menggunakan kolam identitas .....	846
Peran IAM pengguna .....	848
Identitas yang diautentikasi dan tidak diautentikasi .....	848
Aktifkan atau nonaktifkan akses tamu .....	848
Mengubah peran yang terkait dengan jenis identitas .....	849
Edit penyedia identitas .....	851
Menghapus kumpulan identitas .....	852
Menghapus identitas dari kumpulan identitas .....	853
Menggunakan Amazon Cognito Sync dengan kumpulan identitas .....	853
Konsep kolam identitas .....	856
Aliran otentikasi kumpulan identitas .....	857

Peran IAM .....	867
Kepercayaan peran dan izin .....	881
Praktik terbaik keamanan .....	883
Praktik terbaik konfigurasi IAM .....	883
Praktik terbaik konfigurasi kumpulan identitas .....	885
Menggunakan atribut untuk kontrol akses .....	887
Menggunakan atribut untuk kontrol akses dengan kolam identitas Amazon Cognito .....	888
Menggunakan atribut untuk contoh kebijakan kontrol akses .....	890
Matikan atribut untuk kontrol akses .....	892
Pemetaan penyedia default .....	892
Menggunakan kontrol akses berbasis peran .....	894
Membuat peran untuk pemetaan peran .....	894
Memberikan izin peran masuk .....	895
Menggunakan token untuk menetapkan peran kepada pengguna .....	896
Menggunakan pemetaan berbasis aturan untuk menetapkan peran kepada pengguna .....	897
Token mengklaim untuk digunakan dalam pemetaan berbasis aturan .....	899
Praktik terbaik untuk kontrol akses berbasis peran .....	900
Mendapatkan kredensial .....	901
Mengakses layanan AWS .....	908
Identitas mengumpulkan penyedia identitas eksternal .....	911
Facebook .....	911
Login with Amazon .....	920
Google .....	925
Masuk dengan Apple .....	937
Buka penyedia ID Connect .....	944
Penyedia identitas SAML .....	947
Identitas yang diautentikasi pengembang .....	951
Memahami alur otentikasi .....	951
Tentukan nama penyedia pengembang dan kaitkan dengan kumpulan identitas .....	952
Menerapkan penyedia identitas .....	953
Memperbarui peta login (hanya Android dan iOS) .....	961
Mendapatkan token (sisi server) .....	962
Connect ke identitas sosial yang ada .....	964
Mendukung transisi antar penyedia .....	964
Beralih identitas .....	968
Android .....	968

iOS - Objective-C .....	969
iOS - Swift .....	969
JavaScript .....	970
Unity .....	971
Xamarin .....	971
Amazon Cognito Sync .....	972
Memulai dengan Amazon Cognito Sync .....	973
Menyiapkan kumpulan identitas di Amazon Cognito .....	973
Menyimpan dan menyinkronkan data .....	973
Menyinkronkan data .....	973
Menginisialisasi klien Amazon Cognito Sync .....	974
Memahami kumpulan data .....	976
Membaca dan menulis data dalam kumpulan data .....	978
Menyinkronkan data lokal dengan toko sinkronisasi .....	980
Penanganan callback .....	984
Android .....	984
iOS - Objective-C .....	986
iOS - Swift .....	990
JavaScript .....	993
Unity .....	996
Xamarin .....	999
Sinkronisasi push .....	1001
Membuat aplikasi Amazon Simple Notification Service (Amazon SNS) .....	1002
Aktifkan sinkronisasi push di konsol Amazon Cognito .....	1002
Menggunakan sinkronisasi push di aplikasi Anda: Android .....	1003
Menggunakan push sync di aplikasi Anda: iOS - Objective-C .....	1005
Menggunakan push sync di aplikasi Anda: iOS - Swift .....	1008
Pengaliran Amazon Cognito .....	1011
Peristiwa Amazon Cognito .....	1013
Menggunakan konsol Amazon Cognito .....	1019
Konsol kolam pengguna .....	1020
Konsol kumpulan identitas .....	1022
Keamanan .....	1024
Perlindungan data .....	1025
Enkripsi data .....	1025
Pengelolaan identitas dan akses .....	1026

Audiens .....	1027
Mengautentikasi dengan identitas .....	1028
Mengelola akses menggunakan kebijakan .....	1031
Cara kerja Amazon Cognito dengan IAM .....	1034
Contoh kebijakan berbasis identitas .....	1044
Pemecahan Masalah .....	1049
Menggunakan peran terkait layanan .....	1051
Pencatatan dan pemantauan .....	1056
Biaya pemantauan .....	1057
Melacak kuota dan penggunaan di CloudWatch dan Service Quotas .....	1059
Mencatat panggilan API Amazon Cognito dengan AWS CloudTrail .....	1074
Validasi kepatuhan .....	1101
Ketangguhan .....	1102
Pertimbangan data regional .....	1102
Keamanan infrastruktur .....	1103
Konfigurasi dan analisis kerentanan .....	1104
AWS kebijakan terkelola .....	1104
Pembaruan kebijakan .....	1105
Penandaan sumber daya .....	1108
Sumber daya yang didukung .....	1108
Pembatasan tanda .....	1109
Mengelola tag dengan konsol .....	1109
Contoh AWS CLI .....	1110
Menetapkan tanda .....	1110
Melihat tanda .....	1111
Menghapus tanda .....	1111
Menerapkan tanda saat Anda membuat sumber daya .....	1112
Tindakan API .....	1113
Tindakan API untuk tag kumpulan pengguna .....	1113
Tindakan API untuk tag kumpulan identitas .....	1113
Kuota .....	1115
Memahami kuota tingkat permintaan API .....	1115
Kategorisasi kuota .....	1115
Pengguna Amazon Cognito menggabungkan operasi API dengan penanganan tingkat permintaan khusus .....	1116
Pengguna aktif bulanan .....	1117

---

Mengelola kuota tingkat permintaan API .....	1118
Mengidentifikasi persyaratan kuota .....	1118
Optimalkan tarif permintaan .....	1119
Lacak penggunaan kuota .....	1120
Lacak pengguna aktif bulanan (MAUs) .....	1121
Meminta peningkatan kuota .....	1121
Kuota tarif permintaan kumpulan pengguna .....	1122
Kuota tarif permintaan kumpulan identitas .....	1133
Kuota pada jumlah dan ukuran sumber daya .....	1135
Referensi API .....	1142
Referensi titik akhir kumpulan pengguna .....	1142
Referensi titik akhir UI yang di-host .....	1143
Referensi titik akhir Federasi .....	1152
Hibah OAuth 2.0 .....	1176
Menggunakan PKCE .....	1178
Respons kesalahan UI dan federasi yang dihosting .....	1180
Referensi API kumpulan pengguna .....	1182
Referensi API kumpulan identitas .....	1183
Referensi API sinkronisasi Cognito .....	1183
Riwayat dokumen .....	1184
.....	mcci

# Apa itu Amazon Cognito?

Amazon Cognito adalah platform identitas untuk aplikasi web dan seluler. Ini adalah direktori pengguna, server otentikasi, dan layanan otorisasi untuk token akses dan kredensial OAuth 2.0. AWS Dengan Amazon Cognito, Anda dapat mengautentikasi dan mengotorisasi pengguna dari direktori pengguna bawaan, dari direktori perusahaan Anda, dan dari penyedia identitas konsumen seperti Google dan Facebook.

## Topik

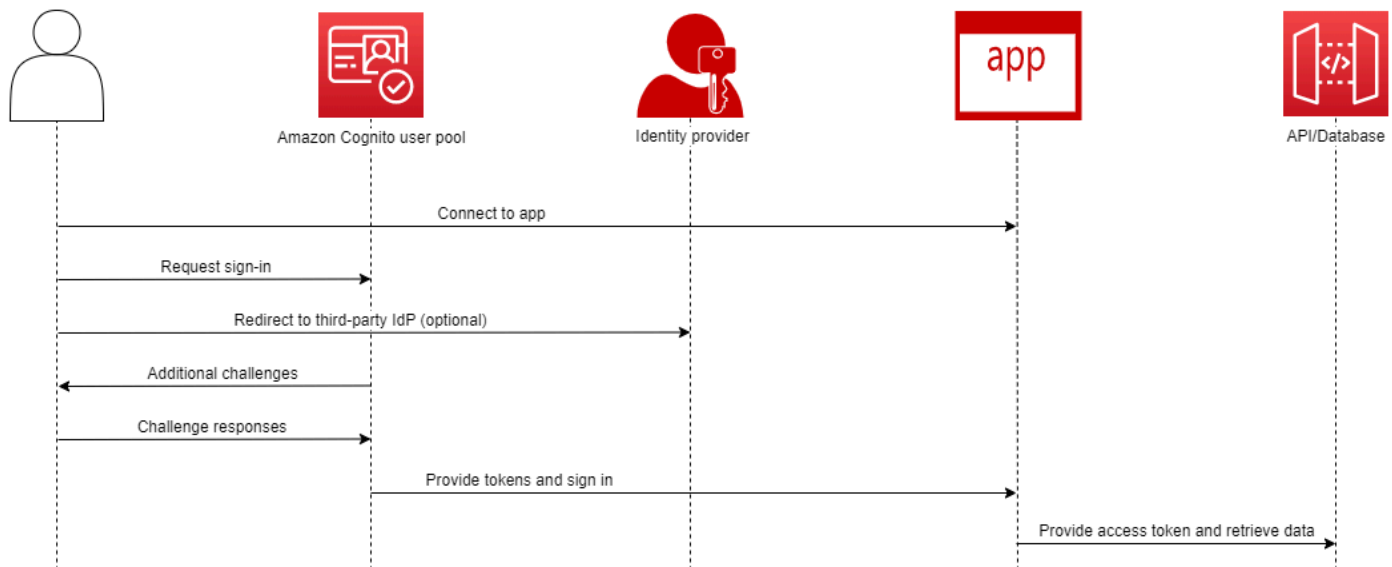
- [Kolam pengguna](#)
- [Kolam identitas](#)
- [Fitur Amazon Cognito](#)
- [Perbandingan kumpulan pengguna Amazon Cognito dan kumpulan identitas](#)
- [Memulai dengan Amazon Cognito](#)
- [Ketersediaan wilayah](#)
- [Harga untuk Amazon Cognito](#)
- [Cara kerja otentikasi dengan kumpulan pengguna Amazon Cognito dan kumpulan identitas](#)
- [Istilah Amazon Cognito](#)
- [Menggunakan layanan ini dengan AWS SDK](#)
- [Memulai dengan AWS](#)

Dua komponen yang mengikuti membentuk Amazon Cognito. Mereka beroperasi secara independen atau bersama-sama, berdasarkan kebutuhan akses Anda untuk pengguna Anda.



# Kolam pengguna

## Amazon Cognito user pools

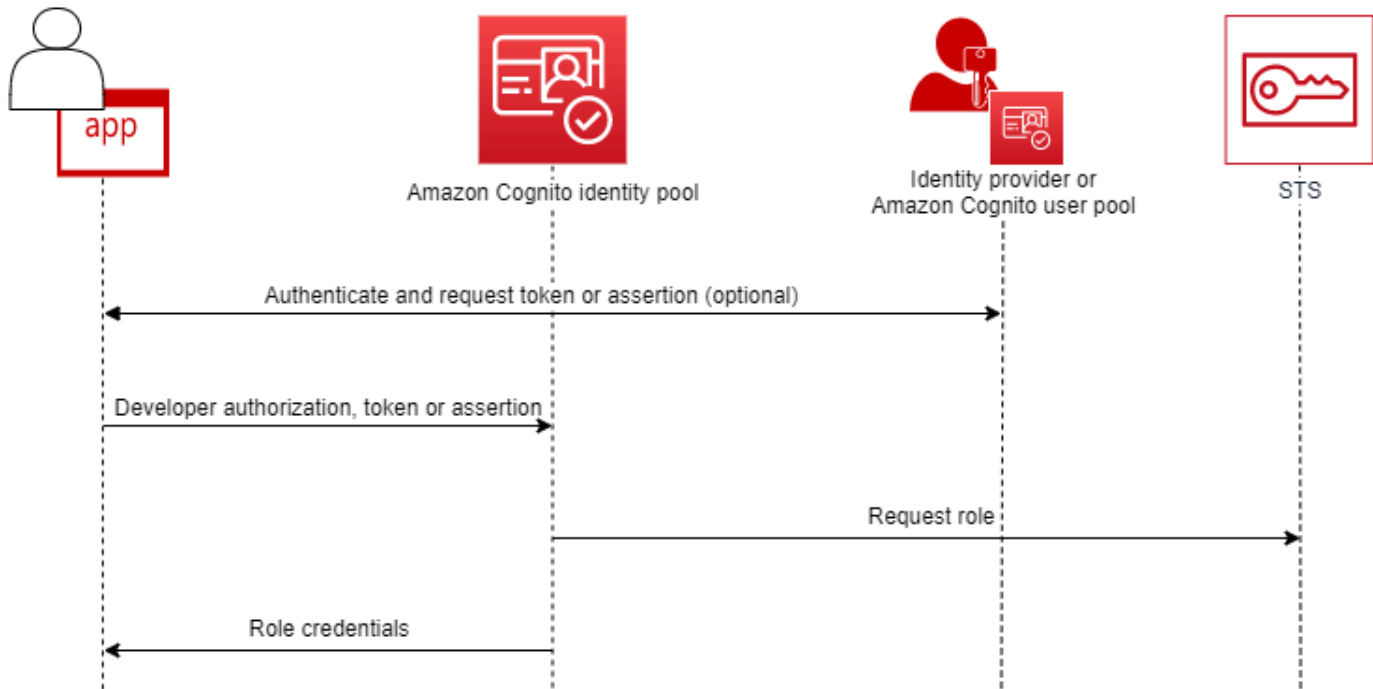


Buat kumpulan pengguna saat Anda ingin mengautentikasi dan mengotorisasi pengguna ke aplikasi atau API Anda. Kumpulan pengguna adalah direktori pengguna dengan pembuatan, manajemen, dan otentikasi pengguna swalayan dan berbasis administrator. Kumpulan pengguna Anda dapat berupa direktori independen dan penyedia identitas OIDC (IDP), dan penyedia layanan perantara (SP) ke penyedia tenaga kerja dan identitas pelanggan pihak ketiga. Anda dapat menyediakan sistem masuk tunggal (SSO) di aplikasi untuk identitas tenaga kerja organisasi Anda di SAMP 2.0 dan OIDC dengan kumpulan pengguna. IdPs Anda juga dapat menyediakan SSO di aplikasi untuk identitas pelanggan organisasi Anda di toko identitas OAuth 2.0 publik Amazon, Google, Apple, dan Facebook. Untuk informasi lebih lanjut tentang identitas pelanggan dan manajemen akses (CIAM), lihat [Apa itu CIAM?](#) .

Kumpulan pengguna tidak memerlukan integrasi dengan kumpulan identitas. Dari kumpulan pengguna, Anda dapat mengeluarkan token web JSON (JWT) yang diautentikasi langsung ke aplikasi, server web, atau API.

## Kolam identitas

### Amazon Cognito federated identities (identity pools)



Siapkan kumpulan identitas Amazon Cognito saat Anda ingin mengotorisasi pengguna yang diautentikasi atau anonim untuk mengakses sumber daya Anda. AWS Kumpulan identitas mengeluarkan AWS kredensial bagi aplikasi Anda untuk menayangkan sumber daya kepada pengguna. Anda dapat mengautentikasi pengguna dengan penyedia identitas tepercaya, seperti kumpulan pengguna atau layanan SAMP 2.0. Ini juga dapat mengeluarkan kredensial secara opsional untuk pengguna tamu. Kumpulan identitas menggunakan kontrol akses berbasis peran dan atribut untuk mengelola otorisasi pengguna Anda untuk mengakses sumber daya Anda. AWS

Identity pool tidak memerlukan integrasi dengan user pool. Kumpulan identitas dapat menerima klaim yang diautentikasi langsung dari penyedia tenaga kerja dan identitas konsumen.

Kumpulan pengguna Amazon Cognito dan kumpulan identitas yang digunakan bersama

Dalam diagram yang memulai topik ini, Anda menggunakan Amazon Cognito untuk mengautentikasi pengguna Anda dan kemudian memberi mereka akses ke file. Layanan AWS

1. Pengguna aplikasi Anda masuk melalui kumpulan pengguna dan menerima token OAuth 2.0.

2. Aplikasi Anda menukar token kumpulan pengguna dengan kumpulan identitas untuk AWS kredensial sementara yang dapat Anda gunakan dengan AWS API dan AWS Command Line Interface (AWS CLI).
3. Aplikasi Anda menetapkan sesi kredensial kepada pengguna Anda, dan memberikan akses resmi seperti Amazon Layanan AWS S3 dan Amazon DynamoDB.

Untuk contoh lainnya yang menggunakan kumpulan identitas dan kumpulan pengguna, lihat [Skenario Amazon Cognito Umum](#).

Di Amazon Cognito, keamanan kewajiban cloud dari [model tanggung jawab bersama](#) sesuai dengan SOC 1-3, PCI DSS, ISO 27001, dan memenuhi syarat HIPAA-BAA. Anda dapat mendesain keamanan Anda di cloud di Amazon Cognito agar sesuai dengan SOC1-3, ISO 27001, dan HIPAA-BAA, tetapi tidak dengan PCI DSS. Untuk informasi selengkapnya, lihat [AWS layanan dalam cakupan](#). Lihat juga [Pertimbangan data regional](#).

## Fitur Amazon Cognito

### Kolam pengguna

Kumpulan pengguna Amazon Cognito adalah direktori pengguna. Dengan kumpulan pengguna, pengguna dapat masuk ke web atau aplikasi seluler Anda melalui Amazon Cognito, atau bergabung melalui iDP pihak ketiga. Pengguna federasi dan lokal memiliki profil pengguna di kumpulan pengguna Anda.

Pengguna lokal adalah mereka yang mendaftar atau Anda buat langsung di kumpulan pengguna Anda. Anda dapat mengelola dan menyesuaikan profil pengguna ini di AWS Management Console, AWS SDK, atau AWS Command Line Interface (AWS CLI).

Kumpulan pengguna Amazon Cognito menerima token dan pernyataan dari pihak ketiga IdPs, dan mengumpulkan atribut pengguna ke dalam JWT yang dikeluarkan ke aplikasi Anda. Anda dapat menstandarisasi aplikasi Anda pada satu set JWT sementara Amazon Cognito menangani interaksi dengan IdPs, memetakan klaim mereka ke format token pusat.

Kumpulan pengguna Amazon Cognito dapat menjadi IDP mandiri. Amazon Cognito mengambil dari standar OpenID Connect (OIDC) untuk menghasilkan JWT untuk otentikasi dan otorisasi. Saat Anda masuk ke pengguna lokal, kumpulan pengguna Anda bersifat otoritatif bagi pengguna tersebut. Anda memiliki akses ke fitur-fitur berikut saat Anda mengautentikasi pengguna lokal.

- Terapkan front-end web Anda sendiri yang memanggil API kumpulan pengguna Amazon Cognito untuk mengautentikasi, mengotorisasi, dan mengelola pengguna Anda.
- Siapkan otentikasi multi-faktor (MFA) untuk pengguna Anda. Amazon Cognito mendukung kata sandi satu kali berbasis waktu (TOTP) dan MFA pesan SMS.
- Aman terhadap akses dari akun pengguna yang berada di bawah kendali jahat.
- Buat alur otentikasi multi-langkah kustom Anda sendiri.
- Cari pengguna di direktori lain dan migrasikan mereka ke Amazon Cognito.

Kumpulan pengguna Amazon Cognito juga dapat memenuhi peran ganda sebagai penyedia layanan (SP) untuk Anda IdPs, dan iDP ke aplikasi Anda. Kumpulan pengguna Amazon Cognito dapat terhubung ke konsumen IdPs seperti Facebook dan Google, atau tenaga kerja IdPs seperti Okta dan Active Directory Federation Services (ADFS).

Dengan token OAuth 2.0 dan OpenID Connect (OIDC) yang bermasalah dengan kumpulan pengguna Amazon Cognito, Anda dapat

- Menerima token ID di aplikasi Anda yang mengautentikasi pengguna, dan memberikan informasi yang Anda perlukan untuk menyiapkan profil pengguna
- Terima token akses di API Anda dengan cakupan OIDC yang mengotorisasi panggilan API pengguna Anda.
- Ambil AWS kredensial dari kumpulan identitas Amazon Cognito.

Fitur kumpulan pengguna Amazon Cognito

Fitur	Deskripsi
IdP OIDC	Menerbitkan token ID untuk mengautentikasi pengguna
Server otorisasi	Keluarkan token akses untuk mengotorisasi akses pengguna ke API
SAML 2.0 SP	Ubah pernyataan SAMP menjadi ID dan token akses
OIDC SP	Ubah token OIDC menjadi ID dan token akses

OAuth 2.0 SP	Ubah token ID dari Apple, Facebook, Amazon, atau Google menjadi ID dan token akses Anda sendiri
Layanan frontend otentikasi	Mendaftar, mengelola, dan mengautentikasi pengguna dengan UI yang dihosting
Dukungan API untuk UI Anda sendiri	Membuat, mengelola, dan mengautentikasi pengguna melalui permintaan API di SDK yang didukung <sup>1</sup> AWS
MFA	Gunakan pesan SMS, TOTP, atau perangkat pengguna Anda sebagai faktor otentikasi tambahan <sup>1</sup>
Pemantauan & respons keamanan	Aman terhadap aktivitas berbahaya dan kata sandi tidak aman <sup>1</sup>
Sesuaikan alur otentikasi	Buat mekanisme otentikasi Anda sendiri, atau tambahkan langkah-langkah khusus ke aliran yang ada <sup>1</sup>
Grup	Buat pengelompokan logis pengguna, dan hierarki klaim peran IAM saat Anda meneruskan token ke kumpulan identitas
Sesuaikan token ID	Sesuaikan token ID Anda dengan klaim baru, dimodifikasi, dan ditekan
Sesuaikan atribut pengguna	Tetapkan nilai ke atribut pengguna dan tambahkan atribut kustom Anda sendiri

<sup>1</sup> Fitur hanya tersedia untuk pengguna lokal.

Untuk informasi selengkapnya tentang kumpulan pengguna, lihat [Memulai dengan kumpulan pengguna](#) referensi [API kumpulan pengguna Amazon Cognito](#).

## Kolam identitas

Kumpulan identitas adalah kumpulan pengenalan unik, atau identitas, yang Anda tetapkan kepada pengguna atau tamu Anda dan diberi wewenang untuk menerima kredensi sementara. AWS Saat Anda menunjukkan bukti otentikasi ke kumpulan identitas dalam bentuk klaim tepercaya dari SAMP 2.0, OpenID Connect (OIDC), atau penyedia identitas sosial OAuth 2.0 (IDP), Anda mengaitkan pengguna Anda dengan identitas di kumpulan identitas. Token yang dibuat oleh kumpulan identitas Anda untuk identitas dapat mengambil kredensial sesi sementara dari AWS Security Token Service (STS).

Untuk melengkapi identitas yang diautentikasi, Anda juga dapat mengonfigurasi kumpulan identitas untuk mengotorisasi akses AWS tanpa autentikasi IDP. Anda dapat menawarkan bukti otentikasi kustom Anda sendiri, atau tidak ada otentikasi. Anda dapat memberikan AWS kredensi sementara kepada pengguna aplikasi mana pun yang memintanya, dengan identitas yang tidak diautentikasi. [Identity pool juga menerima klaim dan mengeluarkan kredensi berdasarkan skema kustom Anda sendiri, dengan identitas yang diautentikasi pengembang.](#)

Dengan kumpulan identitas Amazon Cognito, Anda memiliki dua cara untuk berintegrasi dengan kebijakan IAM di situs Anda. Akun AWS Anda dapat menggunakan kedua fitur ini bersama-sama atau secara individual.

### Kontrol akses berbasis peran

Saat pengguna meneruskan klaim ke kumpulan identitas Anda, Amazon Cognito memilih peran IAM yang diminta. Untuk menyesuaikan izin peran dengan kebutuhan Anda, Anda menerapkan kebijakan IAM untuk setiap peran. Misalnya, jika pengguna Anda menunjukkan bahwa mereka berada di departemen pemasaran, mereka menerima kredensi untuk peran dengan kebijakan yang disesuaikan dengan kebutuhan akses departemen pemasaran. Amazon Cognito dapat meminta peran default, peran berdasarkan aturan yang menanyakan klaim pengguna, atau peran berdasarkan keanggotaan grup pengguna di kumpulan pengguna. Anda juga dapat mengonfigurasi kebijakan kepercayaan peran sehingga IAM hanya mempercayai kumpulan identitas Anda untuk menghasilkan sesi sementara.

### Atribut untuk kontrol akses

Kumpulan identitas Anda membaca atribut dari klaim pengguna Anda, dan memetakannya ke tag utama dalam sesi sementara pengguna Anda. Anda kemudian dapat mengonfigurasi kebijakan berbasis sumber daya IAM Anda untuk mengizinkan atau menolak akses ke sumber daya berdasarkan prinsip IAM yang membawa tag sesi dari kumpulan identitas Anda. Misalnya, jika

pengguna Anda menunjukkan bahwa mereka berada di departemen pemasaran, beri AWS STS tag sesi `Department: marketing` mereka. Bucket Amazon S3 Anda mengizinkan operasi baca berdasarkan `PrincipalTag` kondisi [aws:](#) yang memerlukan nilai `marketing` tag. `Department`

## Fitur kumpulan identitas Amazon Cognito

Fitur	Deskripsi
Kumpulan pengguna Amazon Cognito SP	Tukarkan token ID dari kumpulan pengguna Anda untuk kredensi identitas web dari AWS STS
SAML 2.0 SP	Tukarkan pernyataan SAMP untuk kredensial identitas web dari AWS STS
OIDC SP	Tukar token OIDC untuk kredensial identitas web dari AWS STS
OAuth 2.0 SP	Tukarkan token OAuth dari Amazon, Facebook, Google, Apple, dan Twitter untuk kredensi identitas web dari AWS STS
Kustom SP	Dengan AWS kredensi, tukar klaim dalam format apa pun untuk kredensial identitas web dari AWS STS
Akses tidak diautentikasi	Mengeluarkan kredensial identitas web akses terbatas dari tanpa otentikasi AWS STS
Kontrol akses berbasis peran	Pilih peran IAM untuk pengguna yang diautentikasi berdasarkan klaimnya, dan konfigurasi peran Anda agar hanya diasumsikan dalam konteks kumpulan identitas Anda
Kontrol akses berbasis atribut	Ubah klaim menjadi tag utama untuk sesi AWS STS sementara Anda, dan gunakan kebijakan IAM untuk memfilter akses sumber daya berdasarkan tag utama

Untuk informasi selengkapnya tentang kumpulan identitas, lihat [Memulai dengan kumpulan identitas Amazon Cognito](#) referensi [API kumpulan identitas Amazon Cognito](#).

## Perbandingan kumpulan pengguna Amazon Cognito dan kumpulan identitas

Fitur	Deskripsi	Kolam pengguna	Kolam identitas
IdP OIDC	Menerbitkan token ID OIDC untuk mengautentikasi pengguna aplikasi	✓	
Server otorisasi API	Keluarkan token akses untuk mengotorisasi akses pengguna ke API, database, dan sumber daya lain yang menerima cakupan otorisasi OAuth 2.0	✓	
Server otorisasi identitas web IAM	Hasilkan token yang dapat Anda tukarkan dengan AWS STS AWS kredensi sementara		✓
SAMP 2.0 SP & OIDC IDP	Keluarkan token OIDC yang disesuaikan berdasarkan klaim dari IDP SAMP 2.0	✓	
OIDC SP & OIDC IDP	Mengeluarkan token OIDC yang disesuaikan	✓	



	an berdasarkan klaim dari IDP OIDC	
OAuth 2.0 SP & OIDC IDP	Keluarkan token OIDC yang disesuaikan berdasarkan cakupan dari penyedia sosial OAuth 2.0 seperti Apple dan Google	✓
SAMP 2.0 SP & pialang kredensyal	Menerbitkan AWS kredensi sementara berdasarkan klaim dari IDP SAMP 2.0	✓
OIDC SP & pialang kredensyal	Menerbitkan AWS kredensi sementara berdasarkan klaim dari IDP OIDC	✓
OAuth 2.0 SP & pialang kredensyal	Mengeluarkan AWS kredensi sementara berdasarkan cakupan dari penyedia sosial OAuth 2.0 seperti Apple dan Google	✓
Broker SP & kredensi kumpulan pengguna Amazon Cognito	Menerbitkan AWS kredensi sementara berdasarkan klaim OIDC dari kumpulan pengguna Amazon Cognito	✓
Kustom SP & broker kredensyal	Menerbitkan AWS kredensi sementara berdasarkan otorisasi IAM pengembang	✓

Layanan frontend otentikasi	Mendaftar, mengelola , dan mengautentikasi pengguna dengan UI yang dihosting	✓
Dukungan API untuk UI otentikasi Anda sendiri	Membuat, mengelola , dan mengautentikasi pengguna melalui permintaan API di SDK yang didukung <sup>1</sup> AWS	✓
MFA	Gunakan pesan SMS, TOTP, atau perangkat pengguna Anda sebagai faktor otentikasi tambahan <sup>1</sup>	✓
Pemantauan & respons keamanan	Lindungi dari aktivitas berbahaya dan kata sandi yang tidak aman <sup>1</sup>	✓
Sesuaikan alur otentikasi	Buat mekanisme otentikasi Anda sendiri, atau tambahkan langkah-langkah khusus ke aliran yang ada <sup>1</sup>	✓
Grup	Buat pengelompokan logis pengguna, dan hierarki klaim peran IAM saat Anda meneruskan token ke kumpulan identitas	✓

Sesuaikan token ID	Sesuaikan token ID Anda dengan klaim baru, dimodifikasi, dan ditekan	✓
AWS WAF ACL web	Memantau dan mengontrol permintaan ke lingkungan otentikasi Anda dengan AWS WAF	✓
Sesuaikan atribut pengguna	Tetapkan nilai ke atribut pengguna dan tambahkan atribut kustom Anda sendiri	✓
Akses tidak diautentikasi	Mengeluarkan kredensial identitas web akses terbatas dari tanpa otentikasi AWS STS	✓
Kontrol akses berbasis peran	Pilih peran IAM untuk pengguna yang diautentikasi berdasarkan klaimnya, dan konfigurasi peran Anda agar hanya diasumsikan dalam konteks kumpulan identitas Anda	✓

Kontrol akses berbasis atribut	Ubah klaim pengguna menjadi tag utama untuk sesi AWS STS sementara Anda, dan gunakan kebijakan IAM untuk memfilter akses sumber daya berdasarkan tag utama	✓
--------------------------------	------------------------------------------------------------------------------------------------------------------------------------------------------------	---

<sup>1</sup> Fitur hanya tersedia untuk pengguna lokal.

## Memulai dengan Amazon Cognito

Misalnya aplikasi kumpulan pengguna, lihat [Memulai dengan kumpulan pengguna](#).

Untuk pengenalan kumpulan identitas, lihat [Memulai dengan kumpulan identitas Amazon Cognito](#).

Untuk tautan ke pengalaman penyiapan terpandu dengan kumpulan pengguna dan kumpulan identitas, lihat [Opsis penyiapan terpandu untuk Amazon Cognito](#).

Untuk video, artikel, dokumentasi, dan contoh aplikasi lainnya, lihat sumber daya [pengembang Amazon Cognito](#).

Untuk menggunakan Amazon Cognito, Anda memerlukan file. Akun AWS Untuk informasi selengkapnya, lihat [Memulai dengan AWS](#).

## Ketersediaan wilayah

Amazon Cognito tersedia di beberapa AWS Wilayah di seluruh dunia. Di setiap Wilayah, Amazon Cognito didistribusikan di beberapa Availability Zone. Availability Zone ini secara fisik terisolasi satu sama lain, tetapi disatukan oleh koneksi jaringan privat, latensi rendah, throughput tinggi, dan sangat redundan. Availability Zone ini memungkinkan AWS untuk menyediakan layanan, termasuk Amazon Cognito, dengan tingkat ketersediaan dan redundansi yang sangat tinggi, sekaligus meminimalkan latensi.

Untuk daftar semua Wilayah tempat Amazon Cognito saat ini tersedia, lihat [AWS wilayah dan titik akhir](#) di. Referensi Umum Amazon Web Untuk mempelajari lebih lanjut tentang jumlah Availability Zone yang tersedia di setiap Wilayah, lihat [infrastruktur AWS global](#).

## Harga untuk Amazon Cognito

Untuk informasi tentang harga Amazon Cognito, lihat harga [Amazon Cognito](#).

## Cara kerja otentikasi dengan kumpulan pengguna Amazon Cognito dan kumpulan identitas

Saat pelanggan Anda masuk ke kumpulan pengguna Amazon Cognito, aplikasi Anda menerima token web JSON (JWT).

Saat pelanggan Anda masuk ke kumpulan identitas, baik dengan token kumpulan pengguna atau penyedia lain, aplikasi Anda menerima AWS kredensi sementara.

Dengan login kumpulan pengguna, Anda dapat menerapkan autentikasi dan otorisasi sepenuhnya dengan SDK. AWS Jika Anda tidak ingin membangun komponen antarmuka pengguna (UI) Anda sendiri, Anda dapat memanggil UI web bawaan (UI yang dihosting) atau halaman masuk untuk penyedia identitas pihak ketiga (iDP) Anda.

Topik ini adalah ikhtisar dari beberapa cara aplikasi Anda dapat berinteraksi dengan Amazon Cognito untuk mengautentikasi dengan token ID, mengotorisasi dengan token akses, dan mengakses Layanan AWS dengan kredensial kumpulan identitas.

Topik

- [Autentikasi dan otorisasi API kumpulan pengguna dengan SDK AWS](#)
- [Autentikasi kumpulan pengguna dengan UI yang dihosting](#)
- [Autentikasi kumpulan pengguna dengan penyedia identitas pihak ketiga](#)
- [Autentikasi kumpulan identitas](#)

## Autentikasi dan otorisasi API kumpulan pengguna dengan SDK AWS

AWS telah mengembangkan komponen untuk kumpulan pengguna Amazon Cognito, atau penyedia identitas Amazon Cognito, [dalam berbagai kerangka kerja pengembang](#). Metode yang dibangun

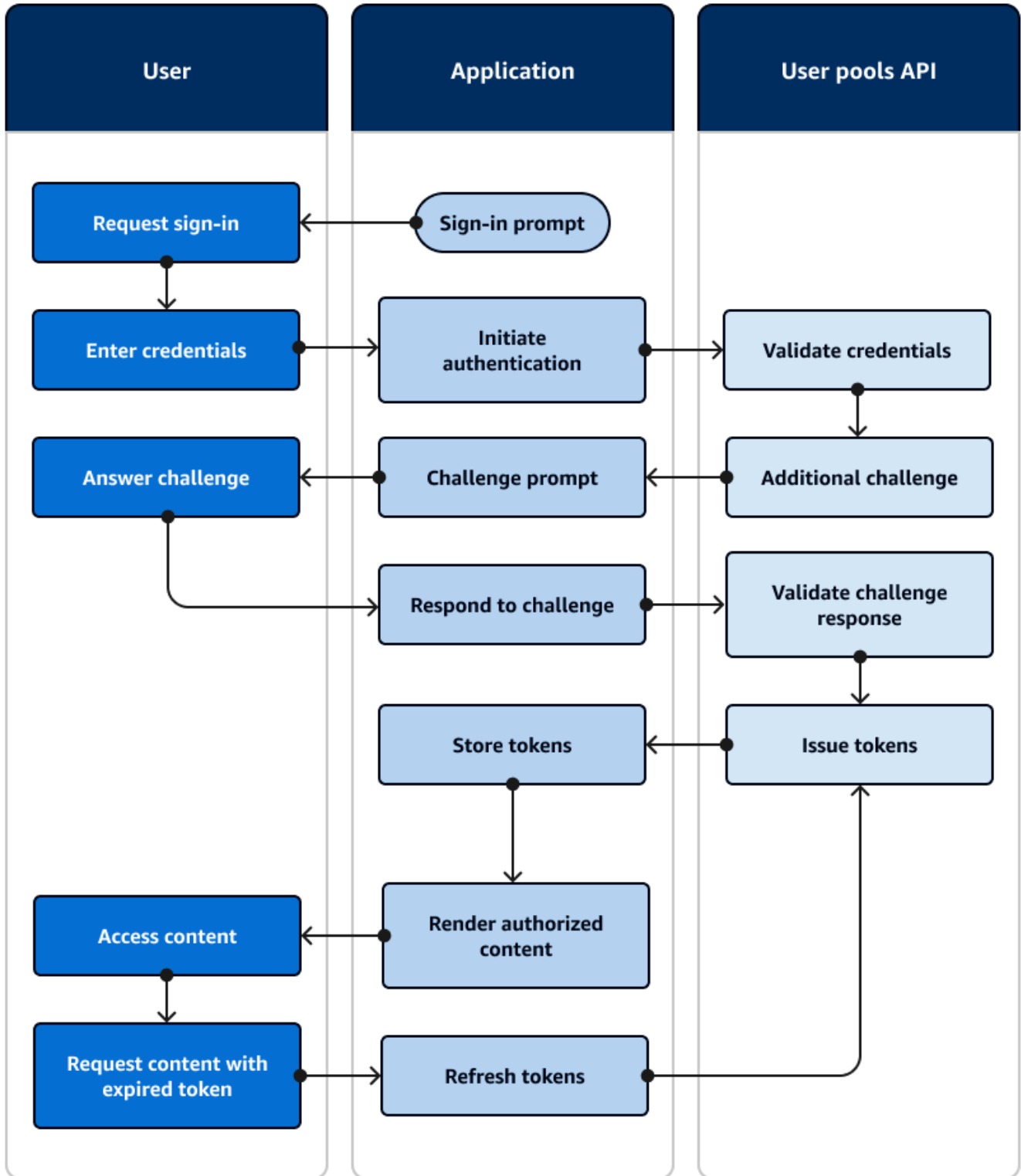
ke dalam SDK ini memanggil API kumpulan [pengguna Amazon Cognito](#). Namespace API pool pengguna yang sama memiliki operasi untuk konfigurasi kumpulan pengguna dan untuk otentikasi pengguna. Untuk ikhtisar yang lebih menyeluruh, lihat [Menggunakan API kumpulan pengguna Amazon Cognito dan titik akhir kumpulan pengguna](#).

Autentikasi API sesuai dengan model di mana aplikasi Anda memiliki komponen UI yang ada dan terutama bergantung pada kumpulan pengguna sebagai direktori pengguna. Desain ini menambahkan Amazon Cognito sebagai komponen dalam aplikasi yang lebih besar. Ini membutuhkan logika terprogram untuk menangani rantai tantangan dan respons yang kompleks.

Aplikasi ini tidak perlu mengimplementasikan implementasi pihak yang mengandalkan OpenID Connect (OIDC) penuh. Sebaliknya, ia memiliki kemampuan untuk memecahkan kode dan menggunakan JWT. Bila Anda menginginkan akses ke set lengkap fitur kumpulan pengguna untuk [pengguna lokal](#), buat autentikasi Anda dengan Amazon Cognito SDK di lingkungan pengembangan Anda.

Otentikasi API dengan cakupan OAuth khusus kurang berorientasi pada otorisasi API eksternal. Untuk menambahkan cakupan khusus ke token akses dari otentikasi API, ubah token saat runtime dengan file. [Pemicu Lambda generasi pra token](#)

Diagram berikut mengilustrasikan sesi masuk khas untuk otentikasi API.



## Alur otentikasi API

1. Seorang pengguna mengakses aplikasi Anda.
2. Mereka memilih tautan “Masuk”.
3. Mereka memasukkan nama pengguna dan kata sandi mereka.
4. Aplikasi memanggil metode yang membuat permintaan [InitiateAuth](#) API. Permintaan meneruskan kredensi pengguna ke kumpulan pengguna.
5. Kumpulan pengguna memvalidasi kredensi pengguna dan menentukan bahwa pengguna telah mengaktifkan otentikasi multi-faktor (MFA).
6. Kumpulan pengguna merespons dengan tantangan yang meminta kode MFA.
7. Aplikasi menghasilkan prompt yang mengumpulkan kode MFA dari pengguna.
8. Aplikasi memanggil metode yang membuat permintaan [RespondToAuthChallenge](#) API. Permintaan melewati kode MFA pengguna.
9. Kumpulan pengguna memvalidasi kode MFA pengguna.
10. Kumpulan pengguna merespons dengan JWT pengguna.
11. Aplikasi menerjemahkan, memvalidasi, dan menyimpan atau menyimpan JWT pengguna.
12. Aplikasi menampilkan komponen yang dikontrol akses yang diminta.
13. Pengguna melihat konten mereka.
14. Kemudian, token akses pengguna telah kedaluwarsa, dan mereka meminta untuk melihat komponen yang dikendalikan akses.
15. Aplikasi menentukan bahwa sesi pengguna harus bertahan. Ini memanggil [InitiateAuth](#) metode lagi dengan token penyegaran dan mengambil token baru.

## Varian dan kustomisasi

Anda dapat menambah alur ini dengan tantangan tambahan—misalnya, tantangan autentikasi kustom Anda sendiri. Anda dapat secara otomatis membatasi akses untuk pengguna yang kata sandinya telah disusupi, atau yang karakteristik login yang tidak terduga mungkin menunjukkan upaya masuk yang berbahaya. Alur ini terlihat hampir sama untuk operasi untuk mendaftar, memperbarui atribut pengguna, dan mengatur ulang kata sandi. Sebagian besar alur ini memiliki operasi API publik duplikat (sisi klien) dan rahasia (sisi server).

## Sumber daya terkait

- [API kumpulan pengguna Amazon Cognito](#)



- [Memulai dengan kumpulan pengguna](#)
- [Mengintegrasikan otentikasi dan otorisasi Amazon Cognito dengan aplikasi web dan seluler](#)
- [Menggunakan API kumpulan pengguna Amazon Cognito dan titik akhir kumpulan pengguna](#)

## Autentikasi kumpulan pengguna dengan UI yang dihosting

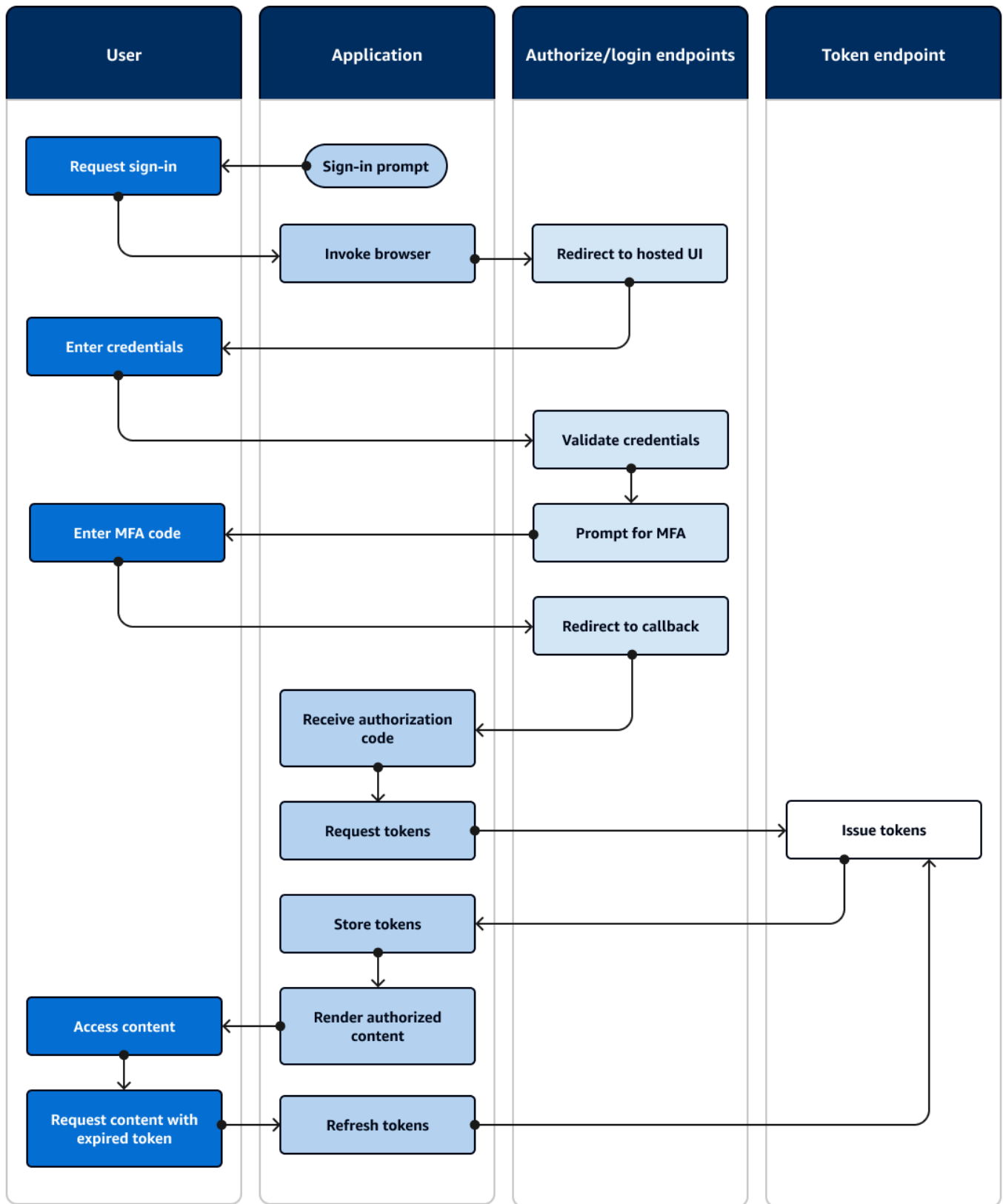
[UI yang dihosting](#) adalah situs web yang ditautkan ke kumpulan pengguna dan klien aplikasi Anda. Ini dapat melakukan operasi masuk, mendaftar, dan mengatur ulang kata sandi untuk pengguna Anda. Aplikasi dengan komponen UI yang di-host untuk otentikasi dapat memerlukan lebih sedikit upaya pengembang untuk mengimplementasikannya. Aplikasi dapat melewati komponen UI untuk otentikasi dan memanggil UI yang dihosting di browser pengguna.

Aplikasi mengumpulkan JWT pengguna dengan lokasi pengalihan web atau aplikasi. Aplikasi yang mengimplementasikan UI yang dihosting dapat terhubung ke kumpulan pengguna untuk otentikasi seolah-olah mereka adalah iDP OpenID Connect (OIDC).

Otentikasi UI yang dihosting sesuai dengan model di mana aplikasi memerlukan server otorisasi, tetapi tidak memerlukan fitur seperti otentikasi khusus, integrasi kumpulan identitas, atau layanan mandiri atribut pengguna. Bila Anda ingin menggunakan beberapa opsi lanjutan ini, Anda dapat menerapkannya dengan komponen kumpulan pengguna untuk SDK.

UI yang dihosting dan model otentikasi IDP pihak ketiga, dengan ketergantungan utama pada implementasi OIDC, adalah yang terbaik untuk model otorisasi lanjutan dengan cakupan OAuth 2.0.

Diagram berikut mengilustrasikan sesi masuk khas untuk otentikasi UI yang dihosting.



## Alur otentikasi UI yang dihosting

1. Seorang pengguna mengakses aplikasi Anda.
2. Mereka memilih tautan “Masuk”.
3. Aplikasi mengarahkan pengguna ke prompt masuk UI yang dihosting.
4. Mereka memasukkan nama pengguna dan kata sandi mereka.
5. Kumpulan pengguna memvalidasi kredensi pengguna dan menentukan bahwa pengguna telah mengaktifkan otentikasi multi-faktor (MFA).
6. UI yang dihosting meminta pengguna untuk memasukkan kode MFA.
7. Pengguna memasukkan kode MFA mereka.
8. UI yang dihosting mengarahkan pengguna ke aplikasi.
9. [Aplikasi mengumpulkan kode otorisasi dari parameter permintaan URL yang ditambahkan UI yang di-host ke URL callback.](#)
10. Aplikasi meminta token dengan kode otorisasi.
11. Titik akhir token mengembalikan JWT ke aplikasi.
12. Aplikasi menerjemahkan, memvalidasi, dan menyimpan atau menyimpan JWT pengguna.
13. Aplikasi menampilkan komponen yang dikontrol akses yang diminta.
14. Pengguna melihat konten mereka.
15. Kemudian, token akses pengguna telah kedaluwarsa, dan mereka meminta untuk melihat komponen yang dikendalikan akses.
16. Aplikasi menentukan bahwa sesi pengguna harus bertahan. Ini meminta token baru dari titik akhir token dengan token penyegaran.

## Varian dan kustomisasi

Anda dapat menyesuaikan tampilan dan nuansa UI yang dihosting dengan CSS di [klien aplikasi](#) apa pun. Anda juga dapat [mengonfigurasi klien aplikasi](#) dengan penyedia identitas mereka sendiri, cakupan, akses ke atribut pengguna, dan konfigurasi keamanan lanjutan.

## Sumber daya terkait

- [Menyiapkan dan menggunakan UI yang dihosting Amazon Cognito dan titik akhir federasi](#)
- [Mendaftar dan masuk dengan UI yang dihosting](#)
- [Cakupan, M2M, dan otorisasi API dengan server sumber daya](#)

- [Titik akhir federasi kumpulan pengguna dan referensi UI yang dihosting](#)

## Autentikasi kumpulan pengguna dengan penyedia identitas pihak ketiga

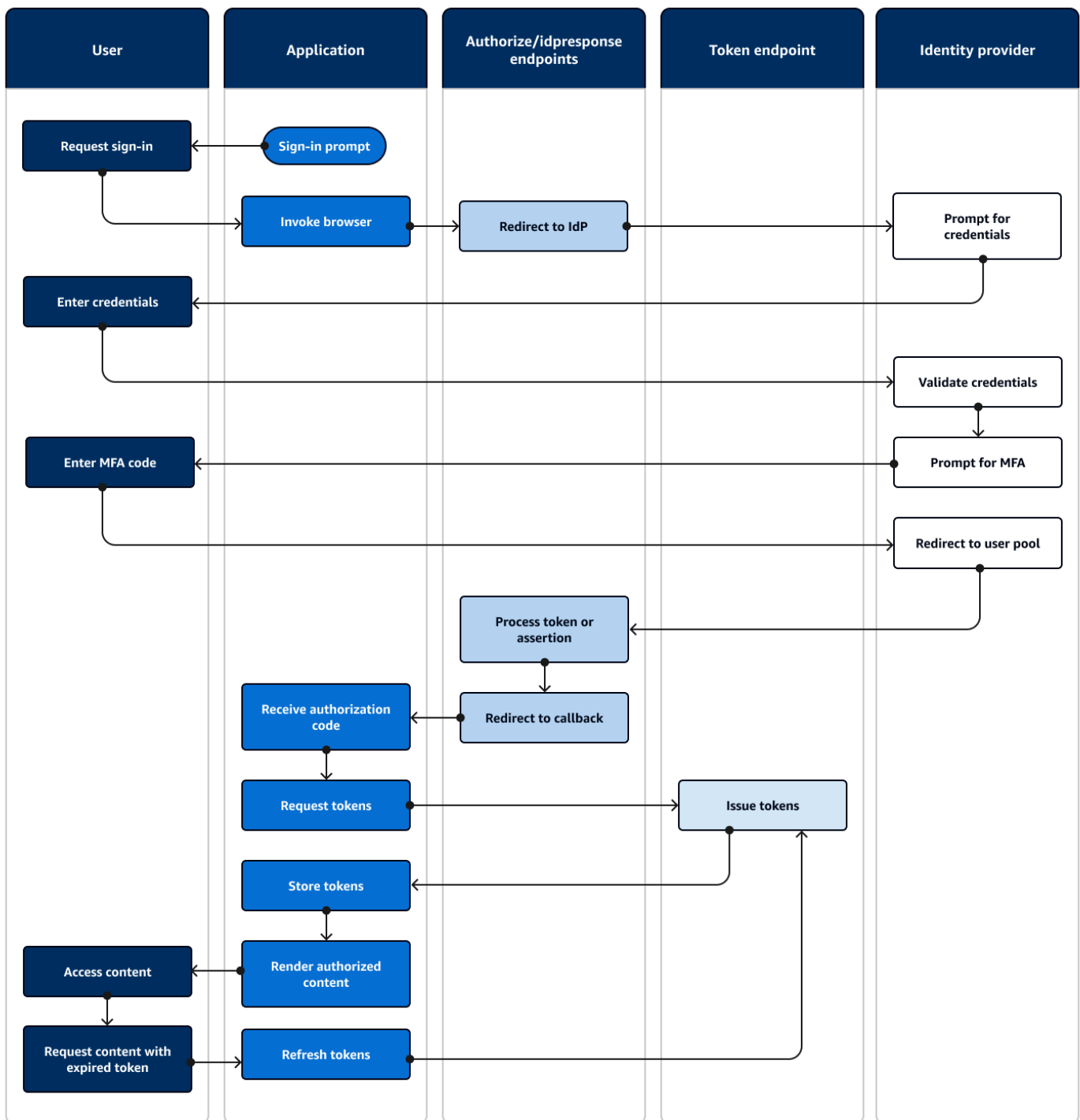
[Masuk dengan penyedia identitas eksternal \(iDP\), atau otentikasi federasi, adalah model yang mirip dengan UI yang dihosting.](#) Aplikasi Anda adalah pihak yang mengandalkan OIDC ke kumpulan pengguna Anda, sementara kumpulan pengguna Anda berfungsi sebagai passthrough ke iDP. IDP dapat berupa direktori pengguna konsumen seperti Facebook atau Google, atau dapat berupa direktori perusahaan SAMP 2.0 atau OIDC seperti Azure.

[Alih-alih UI yang dihosting di browser pengguna, aplikasi Anda memanggil titik akhir pengalihan pada server otorisasi kumpulan pengguna.](#) Dari tampilan pengguna, mereka memilih tombol masuk di aplikasi Anda. Kemudian iDP mereka meminta mereka untuk masuk. Seperti otentikasi UI yang dihosting, aplikasi mengumpulkan JWT di lokasi pengalihan di aplikasi.

Otentikasi dengan iDP pihak ketiga sesuai dengan model di mana pengguna mungkin tidak ingin membuat kata sandi baru saat mereka mendaftar untuk aplikasi Anda. Otentikasi pihak ketiga dapat ditambahkan dengan upaya rendah ke aplikasi yang menerapkan otentikasi UI yang dihosting. Akibatnya, UI yang dihosting dan pihak ketiga IdPs menghasilkan hasil otentikasi yang konsisten dari variasi kecil dalam apa yang Anda panggil di browser pengguna.

Seperti otentikasi UI yang dihosting, otentikasi federasi adalah yang terbaik untuk model otorisasi lanjutan dengan cakupan OAuth 2.0.

Diagram berikut menggambarkan sesi masuk khas untuk otentikasi federasi.



## Alur otentikasi federasi

1. Seorang pengguna mengakses aplikasi Anda.
2. Mereka memilih tautan “Masuk”.

3. Aplikasi mengarahkan pengguna ke prompt masuk dengan IDP mereka.
4. Mereka memasukkan nama pengguna dan kata sandi mereka.
5. IDP memvalidasi kredensi pengguna dan menentukan bahwa pengguna telah mengaktifkan otentikasi multi-faktor (MFA).
6. IDP meminta pengguna untuk memasukkan kode MFA.
7. Pengguna memasukkan kode MFA mereka.
8. IDP mengarahkan pengguna ke kumpulan pengguna dengan respons SAMP atau kode otorisasi.
9. Jika pengguna melewati kode otorisasi, kumpulan pengguna diam-diam menukar kode untuk token iDP. Kumpulan pengguna memvalidasi token IDP dan mengarahkan pengguna ke aplikasi dengan kode otorisasi baru.
- 10 [Aplikasi mengumpulkan kode otorisasi dari parameter permintaan URL yang ditambahkan kumpulan pengguna ke URL callback.](#)
- 11 Aplikasi meminta token dengan kode otorisasi.
12. Titik akhir token mengembalikan JWT ke aplikasi.
- 13 Aplikasi menerjemahkan, memvalidasi, dan menyimpan atau menyimpan JWT pengguna.
- 14 Aplikasi menampilkan komponen yang dikontrol akses yang diminta.
- 15 Pengguna melihat konten mereka.
- 16 Kemudian, token akses pengguna telah kedaluwarsa, dan mereka meminta untuk melihat komponen yang dikendalikan akses.
- 17 Aplikasi menentukan bahwa sesi pengguna harus bertahan. Ini meminta token baru dari titik akhir token dengan token penyegaran.

## Varian dan kustomisasi

[Anda dapat memulai autentikasi federasi di UI yang dihosting, tempat pengguna dapat memilih dari daftar IdPs yang ditetapkan ke klien aplikasi Anda.](#) UI yang dihosting juga dapat meminta alamat email dan [secara otomatis merutekan permintaan pengguna ke IDP](#) SAMP yang sesuai. Otentikasi dengan penyedia identitas pihak ketiga tidak memerlukan interaksi pengguna dengan UI yang dihosting. Aplikasi Anda dapat menambahkan parameter permintaan ke [permintaan server otorisasi](#) pengguna dan menyebabkan pengguna diam-diam mengalihkan ke halaman masuk IDP mereka.

## Sumber daya terkait

- [Menambahkan login kumpulan pengguna melalui pihak ketiga](#)

- [Contoh skenario: bookmark aplikasi Amazon Cognito di dasbor perusahaan](#)
- [Cakupan, M2M, dan otorisasi API dengan server sumber daya](#)
- [Titik akhir federasi kumpulan pengguna dan referensi UI yang dihosting](#)

## Autentikasi kumpulan identitas

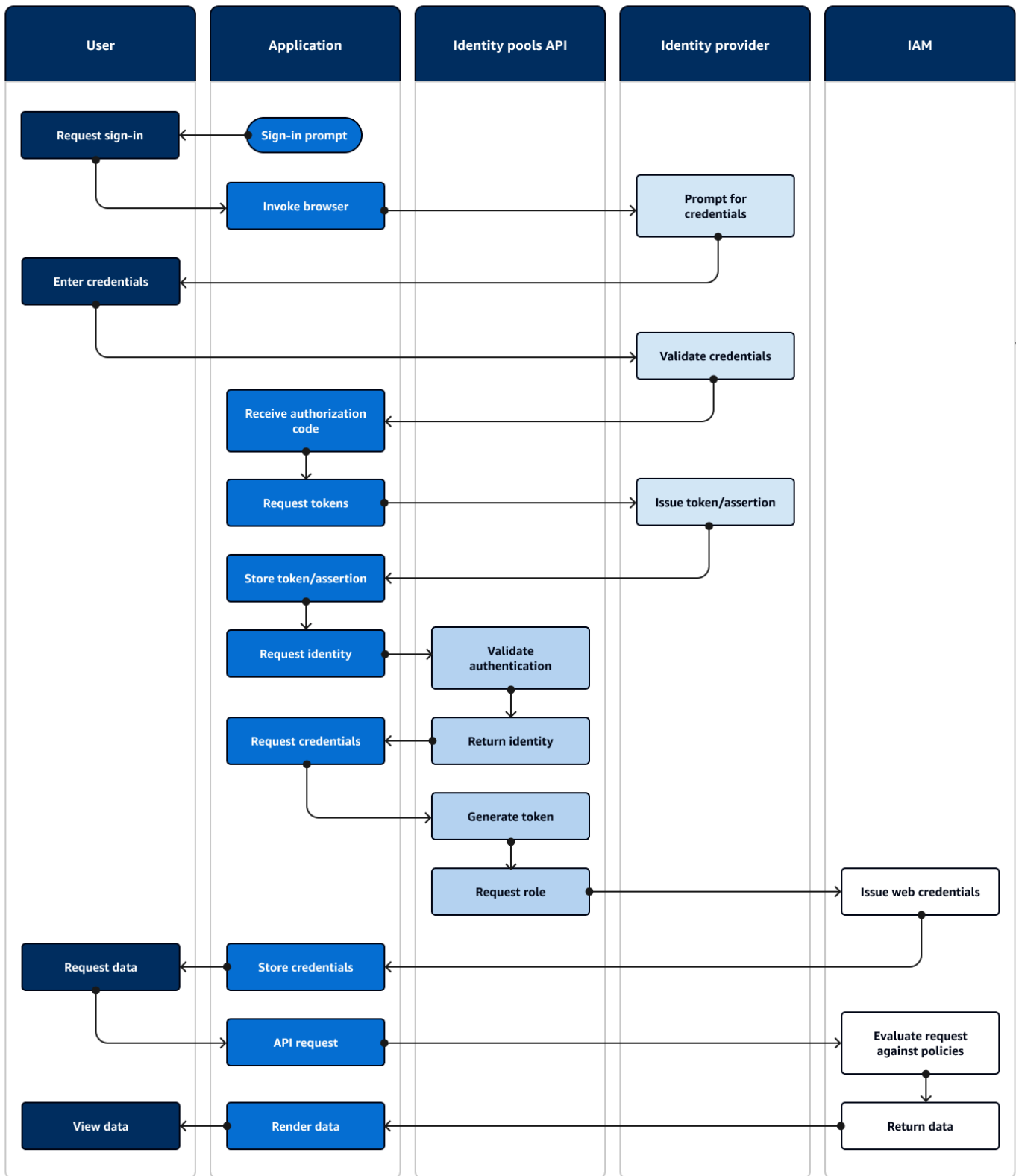
Identity pool adalah komponen untuk aplikasi Anda yang berbeda dari kumpulan pengguna dalam fungsi, namespace API, dan model SDK. Jika kumpulan pengguna menawarkan otentikasi dan otorisasi berbasis token, kumpulan identitas menawarkan otorisasi untuk (IAM). AWS Identity and Access Management

Anda dapat menetapkan satu set kumpulan IdPs identitas dan masuk pengguna dengan mereka. Kumpulan pengguna terintegrasi erat sebagai kumpulan identitas IdPs dan memberikan kolam identitas opsi terbanyak untuk kontrol akses. Pada saat yang sama, ada berbagai pilihan opsi otentikasi untuk kumpulan identitas. Kumpulan pengguna bergabung dengan sumber identitas SAMP, OIDC, sosial, pengembang, dan tamu sebagai rute ke AWS kredensi sementara dari kumpulan identitas.

Otentikasi dengan kumpulan identitas bersifat eksternal—ini mengikuti salah satu alur kumpulan pengguna yang diilustrasikan sebelumnya, atau aliran yang Anda kembangkan secara independen dengan iDP lain. Setelah aplikasi Anda melakukan otentikasi awal, ia meneruskan bukti ke kumpulan identitas dan menerima sesi sementara sebagai imbalan.

Otentikasi dengan kumpulan identitas sesuai dengan model tempat Anda menerapkan kontrol akses untuk aset dan data aplikasi Layanan AWS dengan otorisasi IAM. Seperti [otentikasi API di kumpulan pengguna](#), aplikasi yang berhasil menyertakan AWS SDK untuk setiap layanan yang ingin Anda akses untuk keuntungan pengguna Anda. AWS SDK menerapkan kredensial dari autentikasi kumpulan identitas sebagai tanda tangan ke permintaan API.

Diagram berikut mengilustrasikan sesi masuk khas untuk otentikasi kumpulan identitas dengan IDP.





## Alur otentikasi federasi

1. Seorang pengguna mengakses aplikasi Anda.
2. Mereka memilih tautan “Masuk”.
3. Aplikasi mengarahkan pengguna ke prompt masuk dengan IDP mereka.
4. Mereka memasukkan nama pengguna dan kata sandi mereka.
5. IdP memvalidasi kredensi pengguna.
6. IdP mengarahkan pengguna ke aplikasi dengan respons SAMP atau kode otorisasi.
7. Jika pengguna melewati kode otorisasi, aplikasi menukar kode untuk token iDP.
8. Aplikasi menerjemahkan, memvalidasi, dan menyimpan atau menyimpan JWT atau pernyataan pengguna.
9. Aplikasi memanggil metode yang membuat permintaan [GetId](#) API. Ini melewati token atau pernyataan pengguna dan meminta ID identitas.
10. Kumpulan identitas memvalidasi token atau pernyataan terhadap penyedia identitas yang dikonfigurasi.
11. Identity pool mengembalikan ID identitas.
12. Aplikasi memanggil metode yang membuat permintaan [GetCredentialsForIdentity](#) API. Ini melewati token atau pernyataan pengguna dan meminta peran IAM.
13. Kumpulan identitas menghasilkan JWT baru. JWT baru berisi klaim yang meminta peran IAM. Kumpulan identitas menentukan peran berdasarkan permintaan pengguna dan kriteria pemilihan peran dalam konfigurasi kumpulan identitas untuk iDP.
14. AWS Security Token Service (AWS STS) menanggapi [AssumeRoleWithWebIdentity](#) permintaan dari kumpulan identitas. Respons berisi kredensial API untuk sesi sementara dengan peran IAM.
15. Aplikasi menyimpan kredensial sesi.
16. Pengguna mengambil tindakan di aplikasi yang membutuhkan sumber daya yang dilindungi akses. AWS
17. Aplikasi menerapkan kredensial sementara sebagai [tanda tangan](#) untuk permintaan API untuk yang diperlukan. Layanan AWS
18. IAM mengevaluasi kebijakan yang melekat pada peran dalam kredensial. Ini membandingkannya dengan permintaan.
19. Layanan AWS Mengembalikan data yang diminta.
20. Aplikasi merender data di antarmuka pengguna.

## 21 Pengguna melihat data.

### Varian dan kustomisasi

Untuk memvisualisasikan otentikasi dengan kumpulan pengguna, masukkan salah satu ikhtisar kumpulan pengguna sebelumnya setelah langkah token/ Pernyataan Masalah. Otentikasi pengembang menggantikan semua langkah sebelum Minta identitas dengan permintaan yang ditandatangani oleh kredensial [pengembang](#). [Otentikasi tamu juga melompat langsung ke identitas](#) [Permintaan, tidak memvalidasi otentikasi, dan mengembalikan kredensial untuk peran IAM akses terbatas](#).

### Sumber daya terkait

- [Kumpulan identitas Amazon Cognito](#)
- [Peran IAM pengguna](#)
- [Konsep kolam identitas](#)
- [Aliran otentikasi kumpulan identitas \(identitas federasi\)](#)

## Istilah Amazon Cognito

Amazon Cognito menyediakan kredensial untuk aplikasi web dan seluler. Ini diambil dari dan dibangun berdasarkan istilah-istilah yang umum dalam identitas dan manajemen akses. Banyak panduan untuk identitas universal dan persyaratan akses tersedia. Beberapa contohnya adalah:

- [Terminologi](#) dalam Badan Pengetahuan IDPro
- [AWS Layanan Identitas](#)
- [Glosarium](#) dari NIST CSRC

Daftar berikut menjelaskan istilah yang unik untuk Amazon Cognito atau memiliki konteks tertentu di Amazon Cognito.

### Topik

- [Umum](#)
- [Kolam pengguna](#)
- [Kolam identitas](#)

## Umum

Istilah dalam daftar ini tidak spesifik untuk Amazon Cognito dan diakui secara luas di kalangan praktisi manajemen identitas dan akses. Berikut ini bukan daftar istilah yang lengkap, tetapi panduan untuk konteks Amazon Cognito spesifik mereka dalam panduan ini.

### Aplikasi

Biasanya, aplikasi seluler. Dalam panduan ini, aplikasi sering kali merupakan singkatan untuk aplikasi web atau aplikasi seluler yang terhubung ke Amazon Cognito.

### Kontrol akses berbasis atribut (ABAC)

Model di mana aplikasi menentukan akses ke sumber daya berdasarkan properti pengguna, seperti jabatan atau departemen mereka. Alat Amazon Cognito untuk menerapkan ABAC menyertakan token ID di kumpulan pengguna dan [tag utama](#) di kumpulan identitas.

### Server otorisasi

Sistem berbasis web yang menghasilkan token [web JSON](#). [Titik akhir federasi](#) kumpulan pengguna Amazon Cognito adalah komponen server otorisasi dari dua metode otentikasi dan otorisasi di kumpulan pengguna. Metode lainnya adalah [API kumpulan pengguna](#).

### Aplikasi rahasia, aplikasi sisi server

Aplikasi yang terhubung pengguna dari jarak jauh, dengan kode di server aplikasi dan akses ke rahasia. Ini biasanya aplikasi web.

### Penyedia identitas (iDP)

Layanan yang menyimpan dan memverifikasi identitas pengguna. Amazon Cognito dapat meminta otentikasi dari [penyedia eksternal](#) dan menjadi IDP untuk aplikasi.

### Token web JSON (JWT)

Dokumen berformat JSON yang berisi klaim tentang pengguna yang diautentikasi. Token ID mengautentikasi pengguna, token akses mengotorisasi pengguna, dan menyegarkan kredensi pembaruan token. Amazon Cognito menerima token dari [penyedia eksternal](#) dan mengeluarkan token ke aplikasi atau. AWS STS

### Autentikasi multi-faktor (MFA)

Persyaratan bahwa pengguna memberikan otentikasi tambahan setelah memberikan nama pengguna dan kata sandi mereka. [Kumpulan pengguna Amazon Cognito memiliki fitur MFA untuk pengguna lokal](#).

## Penyedia OAuth 2.0 (sosial)

IdP ke kumpulan pengguna atau kumpulan identitas yang menyediakan akses [JWT](#) dan token penyegaran. Kumpulan pengguna Amazon Cognito mengotomatiskan interaksi dengan penyedia sosial setelah pengguna mengautentikasi.

## Penyedia OpenID Connect (OIDC)

IdP ke kumpulan pengguna atau kumpulan identitas yang memperluas spesifikasi [OAuth](#) untuk menyediakan token ID. Kumpulan pengguna Amazon Cognito mengotomatiskan interaksi dengan penyedia OIDC setelah pengguna mengautentikasi.

## Aplikasi publik

Aplikasi yang mandiri pada perangkat, dengan kode yang disimpan secara lokal dan tidak ada akses ke rahasia. Ini biasanya aplikasi seluler.

## Server sumber daya

API dengan kontrol akses. Kumpulan pengguna Amazon Cognito juga menggunakan server sumber daya untuk mendeskripsikan komponen yang mendefinisikan konfigurasi untuk berinteraksi dengan API.

## Kontrol akses berbasis peran (RBAC)

Model yang memberikan akses berdasarkan penunjukan fungsional pengguna. Kumpulan identitas Amazon Cognito mengimplementasikan RBAC dengan diferensiasi antara peran IAM.

## Penyedia layanan (SP), pihak mengandalkan (RP)

Aplikasi yang mengandalkan IDP untuk menegaskan bahwa pengguna dapat dipercaya. Amazon Cognito bertindak sebagai SP untuk eksternal IdPs, dan sebagai IDP untuk SP berbasis aplikasi.

## SamlProvider

IdP ke kumpulan pengguna atau kumpulan identitas yang menghasilkan dokumen pernyataan yang ditandatangani secara digital yang diteruskan pengguna Anda ke Amazon Cognito.

## Pengidentifikasi Unik Universal (UUID)

Label 128-bit yang diterapkan pada suatu objek. Amazon Cognito UUID unik per kumpulan pengguna atau kumpulan identitas.

## Direktori pengguna

Kumpulan pengguna dan atribut mereka yang melayani informasi itu ke sistem lain. Kumpulan pengguna Amazon Cognito adalah direktori pengguna, dan juga alat untuk konsolidasi pengguna dari direktori pengguna eksternal.

## Kolam pengguna

Ketika Anda melihat istilah dalam daftar berikut dalam panduan ini, mereka merujuk ke fitur atau konfigurasi tertentu dari kumpulan pengguna.

### API kumpulan pengguna Amazon Cognito

Satu set operasi API autentikasi dan otorisasi yang dapat Anda tambahkan ke aplikasi Anda dengan SDK. AWS API dapat masuk ke [pengguna lokal dan pengguna](#) yang [ditautkan](#).

### Otentikasi adaptif

Fitur [keamanan canggih](#) yang mendeteksi potensi aktivitas berbahaya dan menerapkan keamanan tambahan ke [profil pengguna](#).

### Fitur keamanan tingkat lanjut

Komponen opsional yang menambahkan alat untuk keamanan pengguna.

### Klien aplikasi

Komponen yang mendefinisikan pengaturan untuk kumpulan pengguna sebagai iDP untuk satu aplikasi.

### URL panggilan balik, mengarahkan URI

Setelan di [klien aplikasi](#) dan parameter dalam permintaan ke [titik akhir federasi](#) kumpulan pengguna. [URL callback adalah tujuan awal bagi pengguna yang diautentikasi di aplikasi Anda.](#)

### Kredensi yang dikompromikan

Fitur [keamanan canggih](#) yang mendeteksi kata sandi pengguna yang mungkin diketahui penyerang, dan menerapkan keamanan tambahan ke profil [pengguna](#).

### Konfirmasi

Proses yang menentukan bahwa prasyarat telah dipenuhi untuk mengizinkan pengguna baru masuk. Konfirmasi biasanya dilakukan melalui alamat email atau [verifikasi](#) nomor telepon.

## Otentikasi kustom

Perpanjangan proses otentikasi dengan pemicu [Lambda](#) yang menentukan tantangan dan respons pengguna tambahan.

## Otentikasi perangkat

Proses otentikasi yang menggantikan [MFA](#) dengan login yang menggunakan ID perangkat terpercaya.

## Penyedia eksternal, penyedia pihak ketiga

IdP yang memiliki hubungan kepercayaan dengan kumpulan pengguna.

## Pengguna federasi

Pengguna di kumpulan pengguna yang diautentikasi oleh [penyedia eksternal](#).

## Titik akhir Federasi

Satu set halaman web pada [domain kumpulan pengguna](#) Anda yang menghosting layanan untuk interaksi dengan IdPs dan aplikasi.

## UI yang Dihosting

Satu set halaman web interaktif pada [domain kumpulan pengguna](#) Anda yang menghosting layanan untuk otentikasi pengguna.

## Pemicu Lambda

Fungsi di mana AWS Lambda kumpulan pengguna dapat secara otomatis memanggil pada titik-titik kunci dalam proses otentikasi pengguna. Anda dapat menggunakan pemicu Lambda untuk menyesuaikan hasil otentikasi.

## Pengguna lokal

[Profil pengguna](#) di [direktori pengguna kumpulan pengguna](#) yang tidak dibuat dengan autentikasi dengan [penyedia eksternal](#).

## Pengguna tertaut

Pengguna dari [penyedia eksternal](#) yang identitasnya digabungkan dengan [pengguna lokal](#).

## Kustomisasi token

Hasil dari [pemicu Lambda](#) generasi pra token yang memodifikasi ID pengguna atau token akses saat runtime.

Kumpulan pengguna, penyedia identitas Amazon Cognito, **cognito-idp**, kumpulan pengguna Amazon Cognito

AWS Sumber daya dengan layanan otentikasi dan otorisasi untuk aplikasi yang bekerja dengan OIDC. IdPs

Domain kolam pengguna

Nama situs web yang Anda tambahkan ke kumpulan pengguna. Domain adalah URL dasar untuk [UI yang dihosting](#) dan [titik akhir federasi](#).

Verifikasi

Proses konfirmasi bahwa pengguna memiliki alamat email atau nomor telepon. Kumpulan pengguna mengirimkan kode ke pengguna yang telah memasukkan alamat email atau nomor telepon baru. Saat mereka mengirimkan kode ke Amazon Cognito, mereka memverifikasi kepemilikan mereka atas tujuan pesan dan dapat menerima pesan tambahan dari kumpulan pengguna. Juga, lihat [konfirmasi](#).

Profil pengguna, akun pengguna

Entri untuk pengguna di [direktori pengguna](#). Semua pengguna memiliki profil di kumpulan pengguna mereka.

## Kolam identitas

Saat Anda melihat istilah dalam daftar berikut dalam panduan ini, istilah tersebut merujuk ke fitur atau konfigurasi kumpulan identitas tertentu.

Atribut untuk kontrol akses

Implementasi [kontrol akses berbasis atribut di kumpulan](#) identitas. Identity pool menerapkan atribut pengguna sebagai tag untuk kredensi pengguna.

Otentikasi dasar (klasik)

Proses otentikasi di mana Anda dapat menyesuaikan permintaan [kredensial pengguna](#).

Identitas diautentikasi developer

[Proses otentikasi yang mengotorisasi kredensial pengguna kumpulan identitas dengan kredensi pengembang](#).

## Kredensi pengembang

Kunci API IAM dari administrator kumpulan identitas.

## Otentikasi yang ditingkatkan

Alur otentikasi yang memilih peran IAM dan menerapkan tag utama sesuai dengan logika yang Anda tentukan dalam kumpulan identitas Anda.

## Identitas

[UUID](#) yang menautkan pengguna aplikasi dan [kredensialnya](#) ke profil mereka di [direktori pengguna](#) eksternal yang memiliki hubungan kepercayaan dengan kumpulan identitas.

Kolam identitas, identitas federasi Amazon Cognito, identitas Amazon Cognito, **cognito-identity**

AWS Sumber daya dengan layanan otentikasi dan otorisasi untuk aplikasi yang menggunakan kredensi [sementara AWS](#).

## Identitas yang tidak diautentikasi

Pengguna yang belum masuk dengan kumpulan identitas IDP. Anda dapat mengizinkan pengguna untuk menghasilkan kredensi pengguna terbatas untuk satu peran IAM sebelum mereka melakukan autentikasi.

## Kredensi pengguna

Kunci AWS API sementara yang diterima pengguna setelah autentikasi kumpulan identitas.

# Menggunakan layanan ini dengan AWS SDK

AWS kit pengembangan perangkat lunak (SDK) tersedia untuk banyak bahasa pemrograman populer. Setiap SDK menyediakan API, contoh kode, dan dokumentasi yang memudahkan developer untuk membangun aplikasi dalam bahasa pilihan mereka.

Dokumentasi SDK	Contoh kode
<a href="#">AWS SDK for C++</a>	<a href="#">AWS SDK for C++ contoh kode</a>
<a href="#">AWS CLI</a>	<a href="#">AWS CLI contoh kode</a>
<a href="#">AWS SDK for Go</a>	<a href="#">AWS SDK for Go contoh kode</a>
<a href="#">AWS SDK for Java</a>	<a href="#">AWS SDK for Java contoh kode</a>



Dokumentasi SDK	Contoh kode
<a href="#">AWS SDK for JavaScript</a>	<a href="#">AWS SDK for JavaScript contoh kode</a>
<a href="#">AWS SDK for Kotlin</a>	<a href="#">AWS SDK for Kotlin contoh kode</a>
<a href="#">AWS SDK for .NET</a>	<a href="#">AWS SDK for .NET contoh kode</a>
<a href="#">AWS SDK for PHP</a>	<a href="#">AWS SDK for PHP contoh kode</a>
<a href="#">AWS Tools for PowerShell</a>	<a href="#">Alat untuk contoh PowerShell kode</a>
<a href="#">AWS SDK for Python (Boto3)</a>	<a href="#">AWS SDK for Python (Boto3) contoh kode</a>
<a href="#">AWS SDK for Ruby</a>	<a href="#">AWS SDK for Ruby contoh kode</a>
<a href="#">AWS SDK for Rust</a>	<a href="#">AWS SDK for Rust contoh kode</a>
<a href="#">AWS SDK untuk SAP ABAP</a>	<a href="#">AWS SDK untuk SAP ABAP contoh kode</a>
<a href="#">AWS SDK for Swift</a>	<a href="#">AWS SDK for Swift contoh kode</a>

### Ketersediaan contoh

Tidak dapat menemukan apa yang Anda butuhkan? Minta contoh kode menggunakan tautan Berikan umpan balik pada bagian bawah halaman ini.

## Memulai dengan AWS

Sebelum Anda mulai bekerja dengan Amazon Cognito, siapkan diri Anda dengan beberapa sumber daya yang diperlukan AWS .

### Mendaftar untuk Akun AWS

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

Untuk mendaftar untuk Akun AWS

1. Buka <https://portal.aws.amazon.com/billing/signup>.

## 2. Ikuti petunjuk online.

Bagian dari prosedur pendaftaran melibatkan tindakan menerima panggilan telepon dan memasukkan kode verifikasi di keypad telepon.

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

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

## Buat pengguna dengan akses administratif

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

Amankan Anda 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.

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

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

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

Buat pengguna dengan akses 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 pengguna.

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

Masuk sebagai pengguna dengan akses administratif

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

Tetapkan akses ke pengguna tambahan

1. Di Pusat Identitas IAM, buat set izin yang mengikuti praktik terbaik menerapkan izin hak istimewa paling sedikit.

Untuk petunjuknya, lihat [Membuat set izin](#) di Panduan AWS IAM Identity Center Pengguna.

2. Tetapkan pengguna ke grup, lalu tetapkan akses masuk tunggal ke grup.

Untuk petunjuk, lihat [Menambahkan grup](#) di Panduan AWS IAM Identity Center Pengguna.

## Memulai dengan kumpulan pengguna

Anda dapat menggunakan panduan di bagian ini untuk membuat sumber daya kumpulan pengguna awal Anda. Untuk step-by-step panduan, mulailah dengan [aplikasi web dasar di lingkungan JavaScript pengembang React](#). Dari sana, Anda dapat terus menambahkan fitur seperti [antarmuka pengguna yang dihosting \(UI yang dihosting\)](#) dan masuk gabungan dengan penyedia identitas [sosial](#) atau [SAMP 2.0](#) eksternal (). IdPs

Saat Anda berupaya memperluas rangkaian fitur dan menggabungkan lebih banyak komponen Amazon Cognito, baca chapter kumpulan [pengguna Amazon Cognito untuk deskripsi lengkap tentang semua yang dapat Anda lakukan dengan kumpulan pengguna](#).

Contoh kumpulan pengguna dan aplikasi di bagian ini menunjukkan integrasi dasar sumber daya aplikasi dengan kumpulan pengguna Amazon Cognito. Kemudian, Anda dapat menyesuaikan kumpulan pengguna Anda untuk menggunakan lebih banyak opsi yang tersedia untuk Anda. Kemudian Anda dapat memperbarui aplikasi Anda untuk mengadopsi API baru dan berinteraksi dengan UI yang dihosting dan IdPs.

Tutorial di bagian ini membuat aplikasi dengan UI kustom dan otentikasi berbasis API dengan SDK. AWS Aplikasi yang Anda buat dengan cara ini ideal untuk mengautentikasi [pengguna lokal](#). Untuk memulai dengan aplikasi dengan UI bawaan, penanganan otomatis beberapa fitur kumpulan pengguna, dan otentikasi [pengguna federasi](#), lanjutkan ke. [Tambahkan klien aplikasi dengan UI yang dihosting](#)

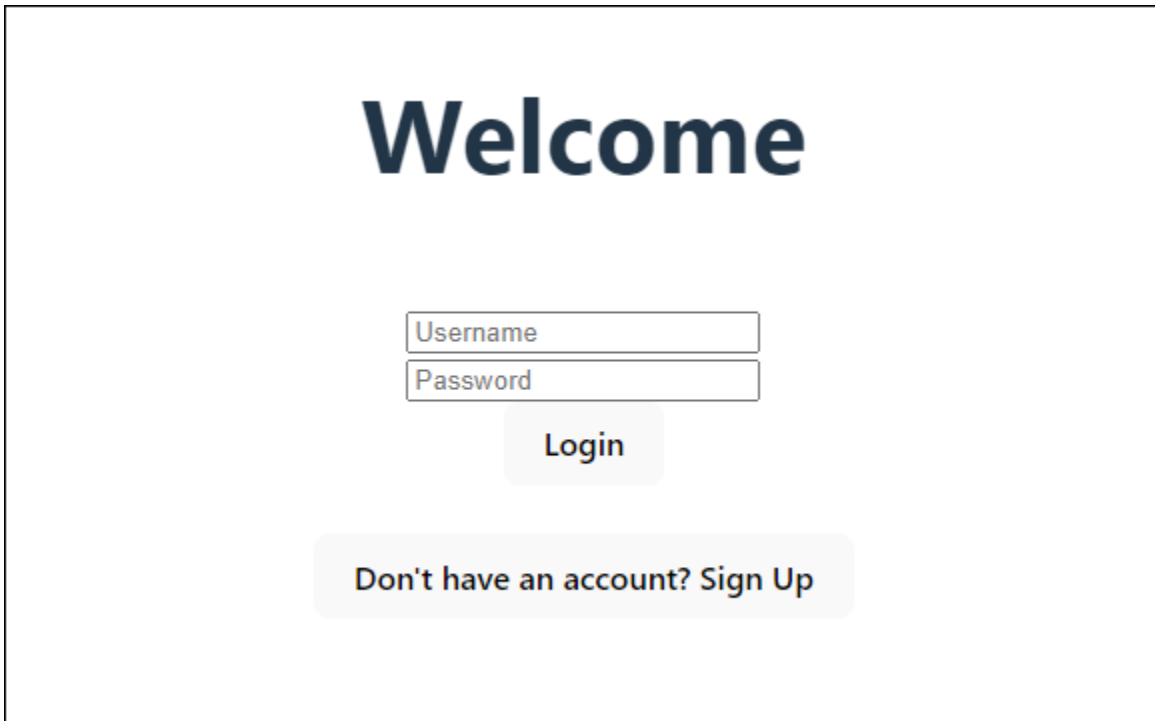
### Topik

- [Siapkan contoh aplikasi satu halaman React](#)
- [Siapkan contoh aplikasi Android dengan Flutter](#)
- [Langkah selanjutnya](#)

## Siapkan contoh aplikasi satu halaman React

Dalam tutorial ini, Anda akan membuat aplikasi satu halaman React di mana Anda dapat menguji sign-up, konfirmasi, dan login pengguna. React adalah perpustakaan JavaScript berbasis untuk aplikasi web dan seluler, dengan fokus pada antarmuka pengguna (UI). Aplikasi contoh ini menunjukkan beberapa fungsi dasar kumpulan pengguna Amazon Cognito. Jika Anda sudah berpengalaman dalam pengembangan aplikasi web dengan React, [unduh contoh aplikasi dari GitHub](#).

Screenshot berikut adalah halaman otentikasi awal dalam aplikasi yang akan Anda buat.



The screenshot shows a login interface. At the top, the word "Welcome" is displayed in a large, bold, dark blue font. Below this, there are two input fields: "Username" and "Password", each with a light gray border. Underneath the input fields is a light gray button with the text "Login" in a bold, dark blue font. At the bottom of the form area, there is a light gray button with the text "Don't have an account? Sign Up" in a bold, dark blue font.

Prosedur [Create a user pool](#) menyiapkan Anda dengan kumpulan pengguna yang bekerja dengan aplikasi contoh. Anda dapat melewati langkah ini jika Anda memiliki kumpulan pengguna yang memenuhi persyaratan berikut:


- Pengguna dapat masuk dengan alamat email mereka. Opsi masuk kumpulan pengguna Cognito: Email.
- Nama pengguna tidak peka huruf besar/kecil. Persyaratan nama pengguna: Membuat nama pengguna sensitif huruf besar/kecil tidak dipilih.
- Otentikasi multi-faktor (MFA) tidak diperlukan. Penegakan MFA: MFA opsional.
- Kumpulan pengguna Anda memverifikasi atribut untuk konfirmasi profil pengguna dengan pesan email. Atribut untuk memverifikasi: Kirim pesan email, verifikasi alamat email.
- Email adalah satu-satunya atribut yang diperlukan. Atribut yang diperlukan: email.
- Pengguna dapat mendaftar sendiri di kumpulan pengguna Anda. Registrasi mandiri: Aktifkan pendaftaran mandiri dipilih.
- Klien aplikasi awal Anda adalah klien publik yang mengizinkan login dengan nama pengguna dan kata sandi. Jenis aplikasi: Klien publik, Alur otentikasi: ALLOW\_USER\_PASSWORD\_AUTH.

## Buat kumpulan pengguna

### Buat kumpulan pengguna baru

1. Masuk ke [Konsol Amazon Cognito](#). Jika diminta, masukkan AWS kredensial Anda.
2. Pilih tombol Buat kumpulan pengguna. Anda mungkin diminta untuk memilih Kumpulan Pengguna dari panel navigasi kiri untuk menampilkan opsi ini.
3. Di sudut kanan atas halaman, pilih Buat kumpulan pengguna untuk memulai panduan pembuatan kumpulan pengguna.
4. Di Mengonfigurasi pengalaman masuk, Anda dapat memilih penyedia identitas (IdPs) yang akan Anda gunakan dengan kumpulan pengguna ini. Untuk informasi selengkapnya, lihat [Menambahkan login kumpulan pengguna melalui pihak ketiga](#).
  - a. Di bawah Penyedia otentikasi, untuk jenis Penyedia, pastikan bahwa hanya kumpulan pengguna Cognito yang dipilih.
  - b. Untuk opsi masuk kumpulan pengguna Cognito, pilih Nama pengguna. Jangan memilih persyaratan nama pengguna tambahan.
  - c. Simpan semua opsi lain sebagai default dan pilih Berikutnya.
5. Di Konfigurasi persyaratan keamanan, Anda dapat memilih kebijakan kata sandi, persyaratan otentikasi multi-faktor (MFA), dan opsi pemulihan akun pengguna. Untuk informasi selengkapnya, lihat [Menggunakan fitur keamanan kumpulan pengguna Amazon Cognito](#).
  - a. Untuk kebijakan Kata Sandi, konfirmasikan bahwa mode kebijakan Kata Sandi disetel ke default Cognito.
  - b. Di bawah otentikasi multi-faktor, untuk penegakan MFA, pilih MFA opsional.
  - c. Untuk metode MFA, pilih Aplikasi Authenticator dan pesan SMS.
  - d. Untuk pemulihan akun Pengguna, konfirmasikan bahwa Aktifkan pemulihan akun swalayan dipilih, dan bahwa metode pengiriman pesan pemulihan akun pengguna disetel ke Email saja.
  - e. Simpan semua opsi lain sebagai default dan pilih Berikutnya.
6. Di Mengonfigurasi pengalaman pendaftaran, Anda dapat menentukan bagaimana pengguna baru akan memverifikasi identitas mereka saat mendaftar sebagai pengguna baru, dan atribut mana yang harus diperlukan atau opsional selama alur pendaftaran pengguna. Untuk informasi selengkapnya, lihat [Mengelola pengguna di kumpulan pengguna Anda](#).

- a. Konfirmasikan bahwa Aktifkan pendaftaran mandiri dipilih. Pengaturan ini membuka kumpulan pengguna Anda untuk mendaftar dari siapa pun di internet. Ini dimaksudkan untuk keperluan aplikasi contoh, tetapi terapkan pengaturan ini dengan hati-hati di lingkungan produksi.
- b. Di bawah Verifikasi dan konfirmasi yang dibantu Cognito, verifikasi bahwa kotak centang Izinkan Cognito mengirim pesan secara otomatis untuk memverifikasi dan mengonfirmasi dipilih.
- c. Konfirmasikan bahwa Atribut yang akan diverifikasi diatur ke Kirim pesan email, verifikasi alamat email.
- d. Di bawah Memverifikasi perubahan atribut, konfirmasikan bahwa opsi default dipilih: Simpan nilai atribut asli saat pembaruan tertunda dipilih, dan Nilai atribut Aktif saat pembaruan tertunda disetel ke Alamat email.
- e. Di bawah atribut Diperlukan, konfirmasikan bahwa atribut Diperlukan berdasarkan pilihan sebelumnya menampilkan email.

 Important

Untuk contoh aplikasi ini, kumpulan pengguna Anda tidak boleh menetapkan `phone_number` sebagai atribut wajib. Jika `phone_number` ditampilkan sebagai atribut wajib, tinjau dan perbarui pilihan Anda sebelumnya:

- MFA opsional, Email hanya untuk metode Pengiriman untuk pesan pemulihan akun pengguna
- Kirim pesan email, verifikasi alamat email untuk Atribut untuk memverifikasi

- f. Simpan semua opsi lain sebagai default dan pilih Berikutnya.
7. Di Konfigurasi pengiriman pesan, Anda dapat mengonfigurasi integrasi dengan Amazon Simple Email Service dan Amazon Simple Notification Service untuk mengirim email dan pesan SMS ke pengguna Anda untuk pendaftaran, konfirmasi akun, MFA, dan pemulihan akun. Untuk informasi selengkapnya, lihat [Pengaturan email untuk kumpulan pengguna Amazon Cognito](#) dan [Pengaturan pesan SMS untuk kolam pengguna Amazon Cognito](#).
- a. Untuk penyedia Email, pilih Kirim email dengan Cognito, dan gunakan pengirim email default yang disediakan oleh Amazon Cognito. Pengaturan untuk volume email rendah ini cukup untuk pengujian aplikasi. Anda dapat kembali setelah memverifikasi alamat email dengan

- Amazon Simple Email Service (Amazon SES) dan memilih Kirim email dengan Amazon SES.
- b. Untuk SMS, pilih Buat peran IAM baru dan masukkan nama peran IAM. Ini menciptakan peran yang memberikan izin ke Amazon Cognito untuk mengirim pesan SMS.
  - c. Simpan semua opsi lain sebagai default dan pilih Berikutnya.
8. Di Integrasikan aplikasi, Anda dapat memberi nama kumpulan pengguna, mengonfigurasi UI yang dihosting, dan membuat klien aplikasi. Untuk informasi selengkapnya, lihat [Tambahkan klien aplikasi dengan UI yang dihosting](#). Contoh aplikasi tidak menggunakan UI yang dihosting.
- a. Di bawah Nama kumpulan pengguna, masukkan nama kumpulan Pengguna.
  - b. Jangan pilih Gunakan UI yang dihosting Cognito.
  - c. Di bawah Klien aplikasi awal, konfirmasikan bahwa jenis Aplikasi disetel ke Klien publik.
  - d. Di bawah rahasia Klien, konfirmasikan bahwa Jangan buat rahasia klien dipilih.
  - e. Masukkan nama klien App.
  - f. Perluas pengaturan klien aplikasi lanjutan. Tambahkan ALLOW\_USER\_PASSWORD\_AUTH ke daftar alur Otentikasi.
  - g. Simpan semua opsi lain sebagai default dan pilih Berikutnya.
9. Tinjau pilihan Anda di Tinjau dan buat layar dan ubah pilihan apa pun sesuai kebutuhan. Jika Anda puas dengan konfigurasi kumpulan pengguna, pilih Buat kumpulan pengguna untuk melanjutkan.
10. Dari halaman kumpulan Pengguna, pilih kumpulan pengguna baru Anda.
11. Di bawah ikhtisar kumpulan pengguna, catat ID kumpulan Pengguna Anda. Anda akan memberikan string ini ketika Anda membuat aplikasi contoh Anda.
12. Pilih tab Integrasi aplikasi dan temukan bagian Klien aplikasi dan analitik. Pilih klien aplikasi baru Anda. Catat ID Klien Anda.

#### Sumber daya terkait

- [Kolam pengguna Amazon Cognito](#)
- [Alur otentikasi kumpulan pengguna](#)
- [Menggunakan token dengan kolam pengguna](#)



## Membuat aplikasi

Untuk membangun aplikasi ini, Anda harus mengatur lingkungan pengembang. Persyaratan lingkungan pengembang adalah:

1. Node.js diinstal dan diperbarui.
2. Manajer paket node (npm) diinstal dan diperbarui ke setidaknya versi 10.2.3.
3. Lingkungan dapat diakses pada port TCP 5173 di browser web.

Untuk membuat contoh aplikasi web React

1. Masuk ke lingkungan pengembang Anda dan arahkan ke direktori induk untuk aplikasi Anda.

```
cd ~/path/to/project/folder/
```

2. Buat layanan React baru.

```
npm create vite@latest frontend-client -- --template react-ts
```

3. Kloning [folder cognito-developer-guide-react-example proyek](#) dari repositori contoh AWS kode pada. GitHub

```
cd ~/some/other/path
```

```
git clone https://github.com/awsdocs/aws-doc-sdk-examples.git
```

```
cp -r ./aws-doc-sdk-examples/javascriptv3/example_code/cognito-identity-provider/scenarios/cognito-developer-guide-react-example/frontend-client ~/path/to/project/folder/frontend-client
```

4. Arahkan ke src direktori di proyek Anda.

```
cd ~/path/to/project/folder/frontend-client/src
```

5. Edit `config.ts` dan ganti nilai-nilai berikut:

- a. Ganti `YOUR_AWS_REGION` dengan Wilayah AWS kode. Misalnya: `us-east-1`.

- b. Ganti `YOUR_COGNITO_USER_POOL_ID` dengan ID kumpulan pengguna yang telah Anda tentukan untuk pengujian. Misalnya: `us-east-1_EXAMPLE`. Kumpulan pengguna harus berada di tempat Wilayah AWS yang Anda masukkan pada langkah sebelumnya.
  - c. Ganti `YOUR_COGNITO_APP_CLIENT_ID` dengan ID klien aplikasi yang telah Anda tentukan untuk pengujian. Misalnya: `1example23456789`. Klien aplikasi harus berada di kumpulan pengguna dari langkah sebelumnya.
6. Jika Anda ingin mengakses aplikasi contoh Anda dari IP selain `localhost`, edit `package.json` dan ubah baris `"dev": "vite"`, ke `"dev": "vite --host 0.0.0.0",`.
  7. Instal aplikasi Anda.

```
npm install
```

8. Luncurkan aplikasi.

```
npm run dev
```

9. Akses aplikasi di browser web di `http://localhost:5173` atau `http://[IP address]:5173`.
10. Daftarkan pengguna baru dengan alamat email yang valid.
11. Ambil kode konfirmasi dari pesan email Anda. Masukkan kode konfirmasi ke dalam aplikasi.
12. Masuk dengan nama pengguna dan kata sandi Anda.

## Membuat lingkungan pengembang React dengan Amazon Lightsail

Cara cepat untuk memulai aplikasi ini adalah dengan membuat server cloud virtual dengan Amazon Lightsail.

Dengan Lightsail, Anda dapat dengan cepat membuat instance server kecil yang telah dikonfigurasi sebelumnya dengan prasyarat untuk aplikasi contoh ini. Anda dapat SSH ke instans Anda dengan klien berbasis browser, dan terhubung ke server web di alamat IP publik atau pribadi.

Untuk membuat instance Lightsail untuk aplikasi contoh ini

1. Pergi ke konsol [Lightsail](#). Jika diminta, masukkan AWS kredensial Anda.
2. Pilih Buat instans.
3. Untuk Pilih platform, pilih Linux/Unix.

4. Untuk Pilih cetak biru, pilih Node.js.
5. Di bawah Identifikasi instans Anda, beri nama yang ramah pada lingkungan pengembangan Anda.
6. Pilih Buat instans.
7. Setelah Lightsail membuat instance Anda, pilih dan dari tab Connect, pilih Connect menggunakan SSH.
8. Sesi SSH terbuka di jendela browser. Jalankan `node -v` dan `npm -v` untuk mengonfirmasi bahwa instance Anda telah disediakan dengan Node.js dan versi npm minimum 10.2.3.
9. Lanjutkan untuk [mengkonfigurasi aplikasi React Anda](#).

## Siapkan contoh aplikasi Android dengan Flutter

Dalam tutorial ini, Anda akan membuat aplikasi seluler di Android Studio tempat Anda dapat meniru perangkat dan menguji pendaftaran, konfirmasi, dan masuk pengguna. Aplikasi contoh ini membuat klien seluler kumpulan pengguna Amazon Cognito dasar untuk Android di Flutter. Jika Anda sudah berpengalaman dalam pengembangan aplikasi seluler dengan Flutter, [unduh aplikasi contoh dari GitHub](#).

Tangkapan layar berikut menunjukkan aplikasi yang berjalan di perangkat Android virtual.

10:06



DEBUG

# Sample Cognito App

Sign-Up

Confirm Sign-Up

Sign-In

## Sign Up

Email

---

Password

---

Sign Up

Prosedur [Create a user pool](#) menyiapkan Anda dengan kumpulan pengguna yang bekerja dengan aplikasi contoh. Anda dapat melewati langkah ini jika Anda memiliki kumpulan pengguna yang memenuhi persyaratan berikut:

- Pengguna dapat masuk dengan alamat email mereka. Opsi masuk kumpulan pengguna Cognito: Email.
- Nama pengguna tidak peka huruf besar/kecil. Persyaratan nama pengguna: Membuat nama pengguna sensitif huruf besar/kecil tidak dipilih.
- Otentikasi multi-faktor (MFA) tidak diperlukan. Penegakan MFA: MFA opsional.
- Kumpulan pengguna Anda memverifikasi atribut untuk konfirmasi profil pengguna dengan pesan email. Atribut untuk memverifikasi: Kirim pesan email, verifikasi alamat email.
- Email adalah satu-satunya atribut yang diperlukan. Atribut yang diperlukan: email.
- Pengguna dapat mendaftar sendiri di kumpulan pengguna Anda. Registrasi mandiri: Aktifkan pendaftaran mandiri dipilih.
- Klien aplikasi awal Anda adalah klien publik yang mengizinkan login dengan nama pengguna dan kata sandi. Jenis aplikasi: Klien publik, Alur otentikasi:ALLOW\_USER\_PASSWORD\_AUTH.

## Buat kumpulan pengguna

### Buat kumpulan pengguna baru

1. Masuk ke [Konsol Amazon Cognito](#). Jika diminta, masukkan AWS kredensial Anda.
2. Pilih tombol Buat kumpulan pengguna. Anda mungkin diminta untuk memilih Kumpulan Pengguna dari panel navigasi kiri untuk menampilkan opsi ini.
3. Di sudut kanan atas halaman, pilih Buat kumpulan pengguna untuk memulai panduan pembuatan kumpulan pengguna.
4. Di Mengonfigurasi pengalaman masuk, Anda dapat memilih penyedia identitas (IdPs) yang akan Anda gunakan dengan kumpulan pengguna ini. Untuk informasi selengkapnya, lihat [Menambahkan login kumpulan pengguna melalui pihak ketiga](#).
  - a. Di bawah Penyedia otentikasi, untuk jenis Penyedia, pastikan bahwa hanya kumpulan pengguna Cognito yang dipilih.
  - b. Untuk opsi masuk kumpulan pengguna Cognito, pilih Nama pengguna. Jangan memilih persyaratan nama pengguna tambahan.
  - c. Simpan semua opsi lain sebagai default dan pilih Berikutnya.

5. Di Konfigurasi persyaratan keamanan, Anda dapat memilih kebijakan kata sandi, persyaratan otentikasi multi-faktor (MFA), dan opsi pemulihan akun pengguna. Untuk informasi selengkapnya, lihat [Menggunakan fitur keamanan kumpulan pengguna Amazon Cognito](#).
  - a. Untuk kebijakan Kata Sandi, konfirmasikan bahwa mode kebijakan Kata Sandi disetel ke default Cognito.
  - b. Di bawah otentikasi multi-faktor, untuk penegakan MFA, pilih MFA opsional.
  - c. Untuk metode MFA, pilih Aplikasi Authenticator dan pesan SMS.
  - d. Untuk pemulihan akun Pengguna, konfirmasikan bahwa Aktifkan pemulihan akun swalayan dipilih, dan bahwa metode pengiriman pesan pemulihan akun pengguna disetel ke Email saja.
  - e. Simpan semua opsi lain sebagai default dan pilih Berikutnya.
6. Di Mengonfigurasi pengalaman pendaftaran, Anda dapat menentukan bagaimana pengguna baru akan memverifikasi identitas mereka saat mendaftar sebagai pengguna baru, dan atribut mana yang harus diperlukan atau opsional selama alur pendaftaran pengguna. Untuk informasi selengkapnya, lihat [Mengelola pengguna di kumpulan pengguna Anda](#).
  - a. Konfirmasikan bahwa Aktifkan pendaftaran mandiri dipilih. Pengaturan ini membuka kumpulan pengguna Anda untuk mendaftar dari siapa pun di internet. Ini dimaksudkan untuk keperluan aplikasi contoh, tetapi terapkan pengaturan ini dengan hati-hati di lingkungan produksi.
  - b. Di bawah Verifikasi dan konfirmasi yang dibantu Cognito, verifikasi bahwa kotak centang Izinkan Cognito mengirim pesan secara otomatis untuk memverifikasi dan mengonfirmasi dipilih.
  - c. Konfirmasikan bahwa Atribut yang akan diverifikasi diatur ke Kirim pesan email, verifikasi alamat email.
  - d. Di bawah Memverifikasi perubahan atribut, konfirmasikan bahwa opsi default dipilih: Simpan nilai atribut asli saat pembaruan tertunda dipilih, dan Nilai atribut Aktif saat pembaruan tertunda disetel ke Alamat email.
  - e. Di bawah atribut Diperlukan, konfirmasikan bahwa atribut Diperlukan berdasarkan pilihan sebelumnya menampilkan email.

**⚠ Important**

Untuk contoh aplikasi ini, kumpulan pengguna Anda tidak boleh menetapkan `phone_number` sebagai atribut wajib. Jika `phone_number` ditampilkan sebagai atribut wajib, tinjau dan perbarui pilihan Anda sebelumnya:

- MFA opsional, Email hanya untuk metode Pengiriman untuk pesan pemulihan akun pengguna
- Kirim pesan email, verifikasi alamat email untuk Atribut untuk memverifikasi

- f. Simpan semua opsi lain sebagai default dan pilih Berikutnya.
7. Di Konfigurasi pengiriman pesan, Anda dapat mengonfigurasi integrasi dengan Amazon Simple Email Service dan Amazon Simple Notification Service untuk mengirim email dan pesan SMS ke pengguna Anda untuk pendaftaran, konfirmasi akun, MFA, dan pemulihan akun. Untuk informasi selengkapnya, lihat [Pengaturan email untuk kumpulan pengguna Amazon Cognito](#) dan [Pengaturan pesan SMS untuk kolam pengguna Amazon Cognito](#).
    - a. Untuk penyedia Email, pilih Kirim email dengan Cognito, dan gunakan pengirim email default yang disediakan oleh Amazon Cognito. Pengaturan untuk volume email rendah ini cukup untuk pengujian aplikasi. Anda dapat kembali setelah memverifikasi alamat email dengan Amazon Simple Email Service (Amazon SES) dan memilih Kirim email dengan Amazon SES.
    - b. Untuk SMS, pilih Buat peran IAM baru dan masukkan nama peran IAM. Ini menciptakan peran yang memberikan izin ke Amazon Cognito untuk mengirim pesan SMS.
    - c. Simpan semua opsi lain sebagai default dan pilih Berikutnya.
  8. Di Integrasikan aplikasi, Anda dapat memberi nama kumpulan pengguna, mengonfigurasi UI yang dihosting, dan membuat klien aplikasi. Untuk informasi selengkapnya, lihat [Tambahkan klien aplikasi dengan UI yang dihosting](#). Contoh aplikasi tidak menggunakan UI yang dihosting.
    - a. Di bawah Nama kumpulan pengguna, masukkan nama kumpulan Pengguna.
    - b. Jangan pilih Gunakan UI yang dihosting Cognito.
    - c. Di bawah Klien aplikasi awal, konfirmasikan bahwa jenis Aplikasi disetel ke Klien publik.
    - d. Di bawah rahasia Klien, konfirmasikan bahwa Jangan buat rahasia klien dipilih.
    - e. Masukkan nama klien App.

- f. Perluas pengaturan klien aplikasi lanjutan. Tambahkan ALLOW\_USER\_PASSWORD\_AUTH ke daftar alur Otentikasi.
  - g. Simpan semua opsi lain sebagai default dan pilih Berikutnya.
9. Tinjau pilihan Anda di Tinjau dan buat layar dan ubah pilihan apa pun sesuai kebutuhan. Jika Anda puas dengan konfigurasi kumpulan pengguna, pilih Buat kumpulan pengguna untuk melanjutkan.
  10. Dari halaman kumpulan Pengguna, pilih kumpulan pengguna baru Anda.
  11. Di bawah ikhtisar kumpulan pengguna, catat ID kumpulan Pengguna Anda. Anda akan memberikan string ini ketika Anda membuat aplikasi contoh Anda.
  12. Pilih tab Integrasi aplikasi dan temukan bagian Klien aplikasi dan analitik. Pilih klien aplikasi baru Anda. Catat ID Klien Anda.

#### Sumber daya terkait

- [Kolam pengguna Amazon Cognito](#)
- [Alur otentikasi kumpulan pengguna](#)
- [Menggunakan token dengan kolam pengguna](#)

## Membuat aplikasi

Untuk membuat contoh aplikasi Android

1. Instal [Android studio](#) dan [alat baris perintah](#).
2. Di Android Studio, instal [plugin Flutter](#).
3. Buat project Android Studio baru dari isi `cognito_flutter_mobile_app` direktori di [aplikasi contoh ini](#).
  - Edit `assets/config.json` <<YOUR USER POOL ID>> dan ganti dan << YOUR CLIENT ID>> dengan ID [kumpulan pengguna dan klien aplikasi yang Anda buat sebelumnya](#).
4. Instal [Flutter](#).
  - a. Tambahkan Flutter ke variabel PATH Anda.
  - b. Terima lisensi dengan perintah berikut.

```
flutter doctor --android-licenses
```



- c. Verifikasi lingkungan Flutter Anda dan instal komponen yang hilang.

```
flutter doctor
```

- Jika ada komponen yang hilang, jalankan `flutter doctor -v` untuk mempelajari cara memperbaiki masalah.

- d. Ubah ke direktori proyek Flutter baru Anda dan instal dependensi.

- Jalankan `flutter pub add amazon_cognito_identity_dart_2`.

- e. Jalankan `flutter pub add flutter_secure_storage`.

5. Buat perangkat Android virtual.

1. Di GUI studio Android, buat perangkat baru dengan [pengelola perangkat](#).

2. Di CLI, jalankan. `flutter emulators --create --name android-device`

6. Luncurkan perangkat Android virtual Anda.

1. Di Android Studio GUI, pilih



ikon mulai di sebelah perangkat virtual Anda.

2. Di CLI, jalankan. `flutter emulators --launch android-device`

7. Luncurkan aplikasi Anda di perangkat virtual Anda.

1. Di GUI Android Studio, pilih ikon



deploy

2. Di CLI, jalankan. `flutter run`

8. Arahkan ke perangkat virtual yang sedang berjalan di Android Studio.

9. Daftarkan pengguna baru dengan alamat email yang valid.

10. Ambil kode konfirmasi dari pesan email Anda. Masukkan kode konfirmasi ke dalam aplikasi.

11. Masuk dengan nama pengguna dan kata sandi Anda.

## Langkah selanjutnya

Setelah Anda mengikuti tutorial untuk menyelesaikan contoh aplikasi, Anda dapat memperluas cakupan implementasi kumpulan pengguna Anda. Anda dapat [membuat kumpulan pengguna](#)

[tambahan](#), [menyesuaikan fitur kumpulan pengguna untuk aplikasi lain](#), atau [menambahkan penyedia identitas eksternal](#). Saat Anda merencanakan langkah Anda untuk menempatkan kumpulan pengguna Amazon Cognito dalam aplikasi produksi, Anda dapat mengevaluasi [contoh dan tutorial tambahan](#).

Berikut ini adalah beberapa fitur kumpulan pengguna Amazon Cognito tambahan:

- [Menyesuaikan halaman web masuk dan daftar bawaan](#)
- [Menambahkan MFA ke kumpulan pengguna](#)
- [Menambahkan keamanan tingkat lanjut ke kumpulan pengguna](#)
- [Menyesuaikan alur kerja kumpulan pengguna dengan pemicu Lambda](#)
- [Menggunakan analitik Amazon Pinpoint dengan kumpulan pengguna Amazon Cognito](#)

Untuk ikhtisar model otentikasi dan otorisasi Amazon Cognito, lihat. [Cara kerja otentikasi dengan kumpulan pengguna Amazon Cognito dan kumpulan identitas](#)

Untuk mengakses lainnya Layanan AWS setelah otentikasi kumpulan pengguna berhasil, lihat [Mengakses Layanan AWS menggunakan kumpulan identitas setelah masuk](#).

Selain menggunakan SDK AWS Management Console dan kumpulan pengguna, Anda juga dapat mengelola kumpulan pengguna Anda dengan menggunakan. [AWS Command Line Interface](#)

Topik

- [Buat kumpulan pengguna baru](#)
- [Tambahkan klien aplikasi dengan UI yang dihosting](#)
- [Tambahkan login sosial ke kumpulan pengguna \(opsional\)](#)
- [Tambahkan login dengan penyedia identitas SAMP ke kumpulan pengguna \(opsional\)](#)

## Buat kumpulan pengguna baru


Dengan kolam pengguna, pengguna Anda dapat masuk ke web atau aplikasi seluler Anda melalui Amazon Cognito.

Buat kumpulan pengguna baru

1. Masuk ke [Konsol Amazon Cognito](#). Jika diminta, masukkan AWS kredensial Anda.

2. Pilih tombol Buat kumpulan pengguna. Anda mungkin diminta untuk memilih Kumpulan Pengguna dari panel navigasi kiri untuk menampilkan opsi ini.
3. Di sudut kanan atas halaman, pilih Buat kumpulan pengguna untuk memulai panduan pembuatan kumpulan pengguna.
4. Di Mengonfigurasi pengalaman masuk, Anda dapat memilih penyedia identitas (IdPs) yang akan Anda gunakan dengan kumpulan pengguna ini. Untuk informasi selengkapnya, lihat [Menambahkan login kumpulan pengguna melalui pihak ketiga](#).
  - a. Di bawah Penyedia otentikasi, untuk jenis Penyedia, pastikan bahwa hanya kumpulan pengguna Cognito yang dipilih.
  - b. Untuk opsi masuk kumpulan pengguna Cognito, pilih Nama pengguna. Jangan memilih persyaratan nama pengguna tambahan.
  - c. Simpan semua opsi lain sebagai default dan pilih Berikutnya.
5. Di Konfigurasi persyaratan keamanan, Anda dapat memilih kebijakan kata sandi, persyaratan otentikasi multi-faktor (MFA), dan opsi pemulihan akun pengguna. Untuk informasi selengkapnya, lihat [Menggunakan fitur keamanan kumpulan pengguna Amazon Cognito](#).
  - a. Untuk kebijakan Kata Sandi, konfirmasikan bahwa mode kebijakan Kata Sandi disetel ke default Cognito.
  - b. Di bawah otentikasi multi-faktor, untuk penegakan MFA, pilih MFA opsional.
  - c. Untuk metode MFA, pilih Aplikasi Authenticator dan pesan SMS.
  - d. Untuk pemulihan akun Pengguna, konfirmasikan bahwa Aktifkan pemulihan akun swalayan dipilih, dan bahwa metode pengiriman pesan pemulihan akun pengguna disetel ke Email saja.
  - e. Simpan semua opsi lain sebagai default dan pilih Berikutnya.
6. Di Mengonfigurasi pengalaman pendaftaran, Anda dapat menentukan bagaimana pengguna baru akan memverifikasi identitas mereka saat mendaftar sebagai pengguna baru, dan atribut mana yang harus diperlukan atau opsional selama alur pendaftaran pengguna. Untuk informasi selengkapnya, lihat [Mengelola pengguna di kumpulan pengguna Anda](#).
  - a. Konfirmasikan bahwa Aktifkan pendaftaran mandiri dipilih. Pengaturan ini membuka kumpulan pengguna Anda untuk mendaftar dari siapa pun di internet. Ini dimaksudkan untuk keperluan aplikasi contoh, tetapi terapkan pengaturan ini dengan hati-hati di lingkungan produksi.

- b. Di bawah Verifikasi dan konfirmasi yang dibantu Cognito, verifikasi bahwa kotak centang Izinkan Cognito mengirim pesan secara otomatis untuk memverifikasi dan mengonfirmasi dipilih.
- c. Konfirmasikan bahwa Atribut yang akan diverifikasi diatur ke Kirim pesan email, verifikasi alamat email.
- d. Di bawah Memverifikasi perubahan atribut, konfirmasikan bahwa opsi default dipilih: Simpan nilai atribut asli saat pembaruan tertunda dipilih, dan Nilai atribut Aktif saat pembaruan tertunda disetel ke Alamat email.
- e. Di bawah atribut Diperlukan, konfirmasikan bahwa atribut Diperlukan berdasarkan pilihan sebelumnya menampilkan email.

 Important

Untuk contoh aplikasi ini, kumpulan pengguna Anda tidak boleh menetapkan `phone_number` sebagai atribut wajib. Jika `phone_number` ditampilkan sebagai atribut wajib, tinjau dan perbarui pilihan Anda sebelumnya:

- MFA opsional, Email hanya untuk metode Pengiriman untuk pesan pemulihan akun pengguna
- Kirim pesan email, verifikasi alamat email untuk Atribut untuk memverifikasi

- f. Simpan semua opsi lain sebagai default dan pilih Berikutnya.
7. Di Konfigurasi pengiriman pesan, Anda dapat mengonfigurasi integrasi dengan Amazon Simple Email Service dan Amazon Simple Notification Service untuk mengirim email dan pesan SMS ke pengguna Anda untuk pendaftaran, konfirmasi akun, MFA, dan pemulihan akun. Untuk informasi selengkapnya, lihat [Pengaturan email untuk kumpulan pengguna Amazon Cognito](#) dan [Pengaturan pesan SMS untuk kolam pengguna Amazon Cognito](#).
- a. Untuk penyedia Email, pilih Kirim email dengan Cognito, dan gunakan pengirim email default yang disediakan oleh Amazon Cognito. Pengaturan untuk volume email rendah ini cukup untuk pengujian aplikasi. Anda dapat kembali setelah memverifikasi alamat email dengan Amazon Simple Email Service (Amazon SES) dan memilih Kirim email dengan Amazon SES.
  - b. Untuk SMS, pilih Buat peran IAM baru dan masukkan nama peran IAM. Ini menciptakan peran yang memberikan izin ke Amazon Cognito untuk mengirim pesan SMS.
  - c. Simpan semua opsi lain sebagai default dan pilih Berikutnya.

8. Di Integrasikan aplikasi, Anda dapat memberi nama kumpulan pengguna, mengonfigurasi UI yang dihosting, dan membuat klien aplikasi. Untuk informasi selengkapnya, lihat [Tambahkan klien aplikasi dengan UI yang dihosting](#). Contoh aplikasi tidak menggunakan UI yang dihosting.
  - a. Di bawah Nama kumpulan pengguna, masukkan nama kumpulan Pengguna.
  - b. Jangan pilih Gunakan UI yang dihosting Cognito.
  - c. Di bawah Klien aplikasi awal, konfirmasikan bahwa jenis Aplikasi disetel ke Klien publik.
  - d. Di bawah rahasia Klien, konfirmasikan bahwa Jangan buat rahasia klien dipilih.
  - e. Masukkan nama klien Aplikasi.
  - f. Perluas pengaturan klien aplikasi lanjutan. Tambahkan ALLOW\_USER\_PASSWORD\_AUTH ke daftar alur Otentikasi.
  - g. Simpan semua opsi lain sebagai default dan pilih Berikutnya.
9. Tinjau pilihan Anda di Tinjau dan buat layar dan ubah pilihan apa pun sesuai kebutuhan. Jika Anda puas dengan konfigurasi kumpulan pengguna, pilih Buat kumpulan pengguna untuk melanjutkan.
10. Dari halaman kumpulan Pengguna, pilih kumpulan pengguna baru Anda.
11. Di bawah ikhtisar kumpulan pengguna, catat ID kumpulan Pengguna Anda. Anda akan memberikan string ini ketika Anda membuat aplikasi contoh Anda.
12. Pilih tab Integrasi aplikasi dan temukan bagian Klien aplikasi dan analitik. Pilih klien aplikasi baru Anda. Catat ID Klien Anda.

Untuk membuat kolam pengguna

1. Masuk ke [Konsol Amazon Cognito](#). Jika diminta, masukkan AWS kredensial Anda.
2. Pilih Kolam Pengguna.
3. Di sudut kanan atas halaman, pilih Buat kumpulan pengguna untuk memulai panduan pembuatan kumpulan pengguna.
4. Di Konfigurasi pengalaman masuk, pilih penyedia gabungan yang akan Anda gunakan dengan kumpulan pengguna ini. Untuk informasi selengkapnya, lihat [Menambahkan login kumpulan pengguna melalui pihak ketiga](#).
5. Di Konfigurasi persyaratan keamanan, pilih kebijakan kata sandi, persyaratan otentikasi multi-faktor (MFA), dan opsi pemulihan akun pengguna. Untuk informasi selengkapnya, lihat [Menggunakan fitur keamanan kumpulan pengguna Amazon Cognito](#).

6. Di Mengonfigurasi pengalaman pendaftaran, tentukan bagaimana pengguna baru akan memverifikasi identitas mereka saat mendaftar, dan atribut mana yang harus diperlukan atau opsional selama alur pendaftaran pengguna. Untuk informasi selengkapnya, lihat [Mengelola pengguna di kumpulan pengguna Anda](#).

#### Important

Jika Anda mengaktifkan pendaftaran pengguna di kumpulan pengguna Anda, siapa pun di internet dapat mendaftar untuk akun dan masuk ke aplikasi Anda. Jangan aktifkan registrasi mandiri di kumpulan pengguna kecuali Anda ingin membuka aplikasi untuk pendaftaran publik. Untuk mengubah setelan ini, perbarui Pendaftaran layanan mandiri di tab Pengalaman pendaftaran konsol kumpulan pengguna, atau perbarui nilai permintaan [AllowAdminCreateUserOnly CreateUserPool](#) atau [UpdateUserPool](#) API. Untuk informasi tentang fitur keamanan yang dapat Anda atur di kumpulan pengguna, lihat [Menggunakan fitur keamanan kumpulan pengguna Amazon Cognito](#).

7. Di Konfigurasi pengiriman pesan, konfigurasi integrasi dengan Amazon Simple Email Service dan Amazon Simple Notification Service untuk mengirim pesan email dan SMS ke pengguna Anda untuk pendaftaran, konfirmasi akun, MFA, dan pemulihan akun. Untuk informasi selengkapnya, lihat [Pengaturan email untuk kumpulan pengguna Amazon Cognito](#) dan [Pengaturan pesan SMS untuk kolam pengguna Amazon Cognito](#).
8. Di Integrasikan aplikasi Anda, beri nama kumpulan pengguna Anda, konfigurasi UI yang dihosting, dan buat klien aplikasi. Untuk informasi selengkapnya, lihat [Tambahkan klien aplikasi dengan UI yang dihosting](#).
9. Tinjau pilihan Anda di Tinjau dan buat layar dan ubah pilihan apa pun sesuai kebutuhan. Jika Anda puas dengan konfigurasi kumpulan pengguna, pilih Buat kumpulan pengguna untuk melanjutkan.

## Sumber daya terkait

Untuk informasi selengkapnya tentang kolam pengguna, lihat [Kolam pengguna Amazon Cognito](#).

Lihat juga: [Alur otentikasi kumpulan pengguna](#) dan [Menggunakan token dengan kolam pengguna](#).

## Tambahkan klien aplikasi dengan UI yang dihosting

Setelah membuat kumpulan pengguna, Anda dapat membuat [klien aplikasi](#) untuk aplikasi yang menampilkan halaman web bawaan UI yang dihosting. Di UI yang dihosting, pengguna dapat:

- Mendaftar untuk profil pengguna.
- Masuk dengan penyedia identitas pihak ketiga.
- Masuk dengan atau tanpa otentikasi multi-faktor.
- Setel ulang kata sandi mereka.

Untuk membuat klien aplikasi untuk login UI yang dihosting

1. Masuk ke [Konsol Amazon Cognito](#). Jika diminta, masukkan AWS kredensial Anda.
2. Pilih Kolam Pengguna.
3. Pilih kolam pengguna yang ada dari daftar, atau [buat kolam pengguna](#). Jika Anda membuat kumpulan pengguna baru, Anda akan diminta untuk menyiapkan klien aplikasi dan mengonfigurasi UI yang dihosting selama wizard.
4. Pilih tab Integrasi aplikasi untuk kumpulan pengguna Anda.
5. Di samping Domain, pilih Tindakan, lalu pilih Buat domain khusus atau Buat domain Amazon Cognito. Jika Anda telah mengonfigurasi domain kumpulan pengguna, pilih Hapus domain Amazon Cognito atau Hapus domain kustom sebelum membuat domain kustom baru Anda.
6. Masukkan awalan domain yang tersedia untuk digunakan dengan domain Amazon Cognito. Untuk informasi tentang cara menyiapkan domain Kustom, lihat [Menggunakan Domain Anda Sendiri untuk UI yang Dihosting](#).
7. Pilih Buat.
8. Arahkan kembali ke tab Integrasi aplikasi untuk kumpulan pengguna yang sama dan temukan klien Aplikasi. Pilih Buat klien aplikasi.
9. Pilih jenis Aplikasi. Beberapa pengaturan yang disarankan akan disediakan berdasarkan pilihan Anda. Aplikasi yang menggunakan UI yang dihosting adalah klien Publik.
10. Masukkan nama klien Aplikasi.
11. Untuk latihan ini, pilih Jangan buat rahasia klien. Rahasia klien digunakan oleh aplikasi rahasia yang mengotentikasi pengguna dari aplikasi terpusat. Dalam latihan ini, Anda akan menampilkan halaman login UI yang dihosting kepada pengguna Anda dan tidak memerlukan rahasia klien.
12. Pilih alur Autentikasi yang akan Anda izinkan dengan aplikasi Anda. Pastikan yang USER\_SRP\_AUTH telah dipilih.

13. Sesuaikan kedaluwarsa token, Konfigurasi keamanan lanjutan, dan izin baca dan tulis Atribut sesuai kebutuhan. Untuk informasi selengkapnya, lihat [Mengonfigurasi Pengaturan Klien Aplikasi](#).
14. Tambahkan URL callback untuk klien aplikasi Anda. Di sinilah Anda akan diarahkan setelah otentikasi UI yang dihosting. Anda tidak perlu menambahkan URL keluar yang Diizinkan hingga Anda dapat menerapkan sign-out di aplikasi.

Untuk aplikasi iOS atau Android, Anda dapat menggunakan URL panggilan balik seperti `myapp://`.

15. Pilih penyedia Identitas untuk klien aplikasi. Minimal, aktifkan kumpulan pengguna Amazon Cognito sebagai penyedia.

#### Note

Untuk masuk dengan penyedia identitas eksternal (IdPs) seperti Facebook, Amazon, Google, dan Apple, serta melalui OpenID Connect (OIDC) atau SAMP IdPs, konfigurasi terlebih dahulu seperti yang ditunjukkan pada [Menambahkan login kumpulan pengguna](#) melalui pihak ketiga. Kemudian kembali ke halaman Pengaturan klien aplikasi untuk mengaktifkannya.

16. Pilih Jenis Hibah OAuth 2.0. Pilih Pemberian kode otorisasi untuk mengembalikan kode otorisasi yang kemudian ditukar dengan token kolam pengguna. Karena tidak pernah berinteraksi langsung ke pengguna akhir, token cenderung tidak dikompromikan. Namun, aplikasi kustom diperlukan pada backend untuk menukar kode otorisasi dengan token kolam pengguna. Untuk alasan keamanan, sebaiknya gunakan alur pemberian kode otorisasi, bersama dengan [Kunci Bukti untuk Pertukaran Kode \(PKCE\)](#), untuk aplikasi seluler.

Pilih Pemberian implisit untuk memerintahkan agar JSON web token (JWT) kolam pengguna kembali kepada Anda dari Amazon Cognito. Anda dapat menggunakan aliran ini ketika tidak ada backend yang tersedia untuk menukar kode otorisasi dengan token. Ini juga membantu untuk debugging token.

#### Note

Anda dapat mengaktifkan baik Pemberian kode otorisasi maupun Pemberian kode implisit, lalu gunakan setiap pemberian sesuai kebutuhan.



Pilih Kredentitas klien hanya jika aplikasi Anda perlu meminta token akses atas namanya sendiri, bukan atas nama pengguna.

17. Kecuali Anda secara khusus ingin mengecualikan satu, pilih semua cakupan OpenID Connect.
18. Pilih cakupan Kustom apa pun yang telah Anda konfigurasi. Cakupan khusus biasanya digunakan dengan klien rahasia.
19. Pilih Buat.

Untuk melihat halaman masuk

Dari halaman klien Aplikasi Anda, pilih Lihat UI yang dihosting untuk membuka tab browser baru ke halaman login yang telah diisi sebelumnya dengan ID klien aplikasi, cakupan, hibah, dan parameter URL callback.

Anda dapat melihat halaman web login UI yang dihosting secara manual dengan URL berikut. Catatan `response_type`. Dalam kasus ini, `response_type=code` untuk pemberian kode otorisasi.

```
https://your_domain/login?  
response_type=code&client_id=your_app_client_id&redirect_uri=your_callback_url
```

Anda dapat melihat halaman web masuk UI yang di-hosting dengan URL berikut untuk pemberian kode implisit jika `response_type=token`. Setelah berhasil masuk, Amazon Cognito mengembalikan token kolam pengguna ke bilah alamat browser web Anda.

```
https://your_domain/login?  
response_type=token&client_id=your_app_client_id&redirect_uri=your_callback_url
```

Anda dapat menemukan token identitas JSON web token (JWT) setelah parameter `#idtoken=` dalam respons.

URL berikut adalah contoh respons dari permintaan hibah implisit. String token identitas Anda akan lebih lama.

```
https://www.example.com/  
#id_token=123456789tokens123456789&expires_in=3600&token_type=Bearer
```

Token kolom pengguna Amazon Cognito ditandatangani dengan algoritme RS256. Anda dapat memecahkan kode dan memverifikasi token kumpulan pengguna menggunakan AWS Lambda. Untuk informasi selengkapnya, lihat [Mendekode dan memverifikasi token Amazon Cognito JWT](#) di situs web. AWS GitHub

Domain Anda ditampilkan di halaman Nama domain. ID klien aplikasi dan URL panggilan balik Anda ditampilkan di halaman Pengaturan umum. Jika perubahan yang Anda buat di konsol tidak segera muncul, tunggu beberapa menit lalu segarkan browser Anda.

## Tambahkan login sosial ke kumpulan pengguna (opsional)

Anda dapat mengaktifkan pengguna aplikasi Anda untuk masuk melalui penyedia identitas (IdP) sosial seperti Facebook, Google, Amazon, dan Apple. Apakah pengguna Anda masuk secara langsung atau melalui pihak ketiga, semua pengguna memiliki profil di kolam pengguna. Lewati langkah ini jika Anda tidak ingin menambahkan masuk melalui penyedia identitas masuk sosial.

### Daftar dengan iDP sosial

Sebelum Anda membuat IdP sosial dengan Amazon Cognito, Anda harus mendaftarkan aplikasi Anda dengan IdP sosial untuk menerima ID klien dan rahasia klien.

Untuk mendaftarkan aplikasi dengan Facebook

1. Buat sebuah [akun developer dengan Facebook](#).
2. [Masuk](#) dengan kredensial Facebook Anda.
3. Dari menu Aplikasi Saya, pilih Buat Aplikasi Baru.

Jika Anda tidak memiliki aplikasi Facebook yang ada, Anda akan melihat opsi yang berbeda. Pilih Buat Aplikasi.

4. Pada halaman Buat aplikasi, pilih kasus penggunaan untuk aplikasi Anda, lalu pilih Berikutnya.
5. Masukkan nama untuk aplikasi Facebook Anda dan pilih Buat Aplikasi.
6. Di bilah navigasi kiri, pilih Pengaturan Aplikasi, lalu pilih Dasar.
7. Perhatikan ID Aplikasi dan Rahasia Aplikasi. Anda akan menggunakannya di bagian selanjutnya.
8. Pilih + Tambahkan platform dari bagian bawah halaman.
9. Pada layar Pilih Platform, pilih platform Anda, lalu pilih Berikutnya.
10. Pilih Simpan perubahan.
11. Untuk Domain Aplikasi, masukkan domain kumpulan pengguna Anda.

```
https://your_user_pool_domain
```

12. Pilih Simpan perubahan.
13. Dari bilah navigasi, pilih Produk, lalu pilih Konfigurasi dari Login Facebook.
14. Dari menu Konfigurasi Login Facebook, pilih Pengaturan.

Masukkan URL pengalihan Anda ke URI Pengalihan OAuth yang Valid. URL pengalihan terdiri dari domain kumpulan pengguna Anda dengan titik `/oauth2/idpresponse` akhir.

```
https://your_user_pool_domain/oauth2/idpresponse
```

15. Pilih Simpan perubahan.

Untuk mendaftarkan aplikasi dengan Amazon

1. Buat sebuah [akun developer dengan Amazon](#).
2. [Masuk](#) dengan kredensial Amazon Anda.
3. Anda perlu untuk membuat profil keamanan Amazon untuk menerima ID klien dan rahasia klien Amazon.

Pilih Aplikasi dan Layanan dari bilah navigasi di bagian atas halaman, lalu pilih Login with Amazon.

4. Pilih Buat Profil Keamanan.
5. Masukkan Nama Profil Keamanan, Deskripsi Profil Keamanan, dan URL Pemberitahuan Privasi Persetujuan.
6. Pilih Simpan.
7. Pilih ID Klien dan Rahasia Klien untuk menampilkan ID dan rahasia klien. Anda akan menggunakannya di bagian selanjutnya.
8. Arahkan kursor ke ikon roda gigi dan pilih Pengaturan Web, lalu pilih Edit.
9. Masukkan domain kumpulan pengguna Anda ke dalam Asal yang Diizinkan.

```
https://<your-user-pool-domain>
```

10. Masukkan domain pool pengguna Anda dengan `/oauth2/idpresponse` titik akhir ke URL Pengembalian yang Diizinkan.

```
https://<your-user-pool-domain>/oauth2/idpresponse
```

## 11. Pilih Simpan.

Untuk mendaftarkan aplikasi dengan Google

Untuk informasi selengkapnya tentang OAuth 2.0 di platform Google Cloud, lihat [Pelajari tentang autentikasi & otorisasi](#) di dokumentasi Google Workspace for Developers.

1. Buat sebuah [akun developer dengan Google](#).
2. Masuk ke [konsol Google Cloud Platform](#).
3. Dari bilah navigasi atas, pilih Pilih proyek. Jika Anda sudah memiliki proyek di platform Google, menu ini menampilkan proyek default Anda sebagai gantinya.
4. Pilih PROYEK BARU.
5. Masukkan nama untuk produk Anda dan kemudian pilih CREATE.
6. Di bilah navigasi kiri, pilih API dan Layanan, lalu pilih layar persetujuan OAuth.
7. Masukkan informasi aplikasi, domain Aplikasi, domain Resmi, dan informasi kontak Pengembang. Domain resmi Anda harus menyertakan `amazoncognito.com` dan akar domain kustom Anda. Misalnya: `example.com`. Pilih SIMPAN DAN LANJUTKAN.
8. 1. Di bawah Cakupan, pilih Tambah atau hapus cakupan, lalu pilih, minimal, cakupan OAuth berikut.
  1. `.../auth/userinfo.email`
  2. `.../auth/userinfo.profile`
  3. `openid`
9. Di bawah Uji pengguna, pilih Tambahkan pengguna. Masukkan alamat email Anda dan pengguna pengujian resmi lainnya, lalu pilih SIMPAN DAN LANJUTKAN.
10. Perluas bilah navigasi kiri lagi, pilih API dan Layanan, lalu pilih Kredensial.
11. Pilih CREATE CREDENTIALS, lalu pilih OAuth client ID.
12. Pilih jenis Aplikasi dan beri nama klien Anda.
13. Di bawah JavaScript Asal resmi, pilih TAMBAH URI. Masukkan domain pool pengguna Anda.

```
https://<your-user-pool-domain>
```

14. Di bawah URI pengalihan resmi, pilih TAMBAH URI. Masukkan jalur ke /oauth2/idpresponse titik akhir domain kumpulan pengguna Anda.

```
https://<your-user-pool-domain>/oauth2/idpresponse
```

15. Pilih BUAT.
16. Simpan nilai yang ditampilkan Google dengan aman di bawah ID klien Anda dan rahasia klien Anda. Berikan nilai ini ke Amazon Cognito saat Anda menambahkan Google iDP.

Untuk mendaftarkan aplikasi dengan Apple

Untuk informasi selengkapnya tentang pengaturan Masuk dengan Apple, lihat [Mengonfigurasi Lingkungan Anda untuk Masuk dengan Apple](#) di dokumentasi Pengembang Apple.

1. Buat sebuah [akun developer dengan Apple](#).
2. [Masuk](#) dengan kredensial Apple Anda.
3. Di bilah navigasi kiri, pilih Sertifikat, Pengidentifikasi & Profil.
4. Di bilah navigasi sebelah kiri, pilih Pengidentifikasi.
5. Pada halaman Pengidentifikasi, pilih ikon +.
6. Pada halaman Daftarkan Pengidentifikasi Baru, pilih ID Aplikasi, lalu pilih Lanjutkan.
7. Pada halaman Pilih jenis, pilih Aplikasi, lalu pilih Lanjutkan.
8. Pada halaman Daftarkan ID Aplikasi, lakukan hal berikut:
  1. Di bawah Deskripsi, masukkan deskripsi.
  2. Di bawah Awalan ID Aplikasi, masukkan ID Bundel. Catat nilai di bawah Awalan ID Aplikasi. Anda akan menggunakan nilai ini setelah memilih Apple sebagai penyedia identitas Anda [Langkah 2: Tambahkan iDP sosial ke kolam pengguna Anda](#).
  3. Pada Kemampuan, pilih Masuk dengan Apple, lalu pilih Edit.
  4. Pada halaman Masuk dengan Apple: Konfigurasi ID Aplikasi, pilih untuk mengatur aplikasi sebagai primer atau dikelompokkan dengan ID Aplikasi lain, lalu pilih Simpan.
  5. Pilih Lanjutkan.
9. Pada halaman Konfirmasi ID Aplikasi Anda, pilih Daftarkan.
10. Pada halaman Pengidentifikasi, pilih ikon +.
11. Pada halaman Daftarkan Pengidentifikasi Baru, pilih ID Layanan, lalu pilih Lanjutkan.

12. Pada halaman Daftarkan ID Layanan, lakukan hal berikut:
  1. Di bawah Deskripsi, masukkan deskripsi.
  2. Di bawah Identifier, masukkan pengenalan. Catat ID Layanan ini karena Anda memerlukan nilai ini setelah memilih Apple sebagai penyedia identitas Anda [Langkah 2: Tambahkan iDP sosial ke kolam pengguna Anda](#).
  3. Pilih Lanjutkan, lalu pilih Daftarkan.
13. Pilih ID Layanan yang baru saja Anda buat dari halaman Pengidentifikasi.
  1. Pilih Masuk dengan Apple, lalu pilih Konfigurasi.
  2. Pada halaman Konfigurasi Otentikasi Web, pilih ID aplikasi yang Anda buat sebelumnya sebagai ID Aplikasi Utama.
  3. Pilih ikon+di sebelah URL Situs Web.
  4. Di bawah Domain dan subdomain, masukkan domain kumpulan pengguna Anda tanpa awalan. `https://`

`<your-user-pool-domain>`
  5. Di bawah URL Kembali, masukkan jalur ke `/oauth2/idpresponse` titik akhir domain kumpulan pengguna Anda.

`https://<your-user-pool-domain>/oauth2/idpresponse`
  6. Pilih Berikutnya, lalu pilih Selesai. Anda tidak perlu memverifikasi domain.
  7. Pilih Lanjutkan, lalu pilih Simpan.
14. Di bilah navigasi sebelah kiri, pilih Kunci.
15. Pada halaman Kunci, pilih ikon +.
16. Pada halaman Daftarkan Kunci Baru, lakukan hal berikut:
  1. Di bawah Nama Kunci, masukkan nama kunci.
  2. Pilih Masuk dengan Apple, lalu pilih Konfigurasi.
  3. Pada halaman Configure Key, pilih ID aplikasi yang Anda buat sebelumnya sebagai ID Aplikasi Utama. Pilih Simpan.
  4. Pilih Lanjutkan, lalu pilih Daftarkan.
17. Pada halaman Unduh Kunci Anda, pilih Unduh untuk mengunduh kunci pribadi, catat ID Kunci yang ditampilkan, lalu pilih Selesai. Anda akan membutuhkan kunci privat ini dan nilai ID Kunci

yang ditampilkan di halaman ini setelah Anda memilih Apple sebagai penyedia identitas di [Langkah 2: Tambahkan iDP sosial ke kolam pengguna Anda](#).

## Tambahkan iDP sosial ke kumpulan pengguna Anda

Di bagian ini, Anda mengonfigurasi IdP sosial di kolam pengguna Anda menggunakan ID klien dan rahasia klien dari bagian sebelumnya.

Untuk mengonfigurasi penyedia identitas sosial kumpulan pengguna dengan AWS Management Console

1. Masuk ke [Konsol Amazon Cognito](#). Anda mungkin diminta untuk AWS kredensial Anda.
2. Pilih Kolam Pengguna.
3. Pilih kolam pengguna yang ada dari daftar, atau [buat kolam pengguna](#).
4. Pilih tab Pengalaman masuk. Cari Masuk Federasi dan pilih Tambahkan penyedia identitas.
5. Pilih penyedia identitas sosial: Facebook, Google, Login with Amazon, atau Masuk dengan Apple.
6. Pilih dari langkah-langkah berikut, berdasarkan pilihan penyedia identitas sosial Anda:
  - Google dan Login with Amazon — Masukkan ID klien aplikasi dan rahasia klien aplikasi yang dihasilkan di bagian sebelumnya.
  - Facebook — Masukkan ID klien aplikasi dan rahasia klien aplikasi yang dihasilkan di bagian sebelumnya, lalu pilih versi API (misalnya, versi 2.12). Sebaiknya pilih versi terbaru yang mungkin—setiap API Facebook memiliki siklus hidup dan tanggal penghentian. Cakupan dan atribut Facebook dapat bervariasi antar versi API. Kami merekomendasikan pengujian login identitas sosial Anda dengan Facebook untuk memastikan bahwa federasi berfungsi sebagaimana dimaksud.
  - Masuk dengan Apple — Masukkan ID Layanan, ID Tim, ID Kunci, dan kunci pribadi yang dibuat di bagian sebelumnya.
7. Masukkan nama cakupan Resmi yang ingin Anda gunakan. Cakupan menentukan atribut pengguna (seperti name dan email) mana yang ingin Anda akses dengan aplikasi Anda. Untuk Facebook, ini harus dipisahkan dengan koma. Untuk Google dan Login with Amazon, mereka harus dipisahkan dengan spasi. Untuk Masuk dengan Apple, pilih kotak centang untuk cakupan yang ingin Anda akses.

Penyedia identitas sosial	Contoh cakupan
Facebook	<code>public_profile, email</code>
Google	<code>profile email openid</code>
Login with Amazon	<code>profile postal_code</code>
Masuk dengan Apple	<code>email name</code>

Pengguna aplikasi Anda diminta untuk menyetujui penyediaan atribut ini ke aplikasi Anda. Untuk informasi selengkapnya tentang cakupan penyedia sosial, lihat dokumentasi dari Google, Facebook, Login with Amazon, atau Masuk dengan Apple.

Dengan Masuk dengan Apple, berikut ini adalah skenario pengguna di mana cakupan mungkin tidak dikembalikan:

- Pengguna akhir mengalami kegagalan setelah meninggalkan halaman masuk Apple (ini bisa dari kegagalan internal dalam Amazon Cognito atau apa pun yang ditulis oleh pengembang).
  - ID layanan identifikasi digunakan di seluruh kumpulan pengguna dan/atau layanan otentikasi lainnya.
  - Pengembang menambahkan cakupan tambahan setelah pengguna masuk. Pengguna hanya mengambil informasi baru ketika mereka mengautentikasi dan ketika mereka menyegarkan token mereka.
  - Pengembang menghapus pengguna dan kemudian pengguna masuk lagi tanpa menghapus aplikasi dari profil ID Apple mereka.
8. Petakan atribut dari penyedia identitas Anda ke kumpulan pengguna Anda. Untuk informasi selengkapnya, lihat [Hal yang perlu diketahui tentang pemetaan](#).
  9. Pilih Buat.
  10. Dari tab Integrasi klien aplikasi, pilih salah satu klien Aplikasi dalam daftar dan Edit pengaturan UI yang dihosting. Tambahkan penyedia identitas sosial baru ke klien aplikasi di bawah Penyedia identitas.
  11. Pilih Simpan perubahan.



## Uji konfigurasi iDP sosial Anda

Anda dapat membuat URL login dengan menggunakan elemen dari dua bagian sebelumnya. Gunakan URL login itu untuk menguji konfigurasi IdP sosial Anda.

```
https://mydomain.us-east-1.amazoncognito.com/login?  
response_type=code&client_id=1example23456789&redirect_uri=https://www.example.com
```

Anda dapat menemukan domain Anda di halaman konsol Nama Domain kolam pengguna. `client_id` ada di halaman Pengaturan klien aplikasi. Gunakan URL panggilan balik Anda untuk parameter `redirect_uri`. Ini adalah URL halaman tempat pengguna Anda akan diarahkan setelah autentikasi berhasil.

### Note

Amazon Cognito membatalkan permintaan otentikasi yang tidak selesai dalam 5 menit, dan mengarahkan pengguna ke UI yang dihosting. Halaman menampilkan pesan `Something went wrong` kesalahan.

## Tambahkan login dengan penyedia identitas SAMP ke kumpulan pengguna (opsional)

Anda dapat mengaktifkan pengguna aplikasi Anda untuk masuk melalui penyedia identitas SAML (IdP). Apakah pengguna Anda masuk secara langsung atau melalui pihak ketiga, semua pengguna memiliki profil di kolam pengguna. Lewati langkah ini jika Anda tidak ingin menambahkan masuk melalui penyedia identitas SAML.

Untuk informasi selengkapnya, lihat [Menggunakan penyedia identitas SAMP dengan kumpulan pengguna](#).

Anda harus memperbarui penyedia identitas SAML Anda dan mengonfigurasi kumpulan pengguna Anda. Untuk informasi tentang cara menambahkan kumpulan pengguna Anda sebagai pihak yang mengandalkan atau aplikasi untuk penyedia identitas SAMP 2.0 Anda, lihat dokumentasi untuk penyedia identitas SAMP Anda.

Anda juga harus memberikan titik akhir assertion consumer service (ACS) kepada penyedia identitas SALL Anda. Konfigurasi titik akhir berikut di domain kumpulan pengguna Anda untuk pengikatan

SAMP 2.0 POST di penyedia identitas SAMP Anda. Untuk informasi selengkapnya tentang domain kumpulan pengguna, lihat [Mengkonfigurasi domain kolam pengguna](#).

```
https://Your user pool domain/saml2/idpresponse
```

With an Amazon Cognito domain:

```
https://<yourDomainPrefix>.auth.<region>.amazoncognito.com/saml2/idpresponse
```

With a custom domain:

```
https://Your custom domain/saml2/idpresponse
```

Anda dapat menemukan awalan domain dan nilai Wilayah untuk kumpulan pengguna Anda di tab Nama domain di konsol [Amazon Cognito](#).

Untuk beberapa penyedia identitas SAMP, Anda juga perlu menyediakan penyedia layanan (SP)urn, juga disebut URI audiens atau ID entitas SP, dalam format:

```
urn:amazon:cognito:sp:<yourUserPoolID>
```

Anda dapat menemukan ID kolam pengguna Anda di tab Pengaturan umum di [Konsol Amazon Cognito](#).


Anda juga harus mengonfigurasi penyedia identitas SAML Anda untuk memberikan nilai atribut untuk atribut yang diperlukan di kolam pengguna Anda. Biasanya, email adalah atribut yang diperlukan untuk kolam pengguna. Dalam hal itu, penyedia identitas SAML harus memberikan nilai (klaim) email dalam pernyataan SAML.

Kolam pengguna Amazon Cognito mendukung federasi SAML 2.0 dengan titik akhir pasca-pengikatan. Ini menghilangkan kebutuhan aplikasi Anda untuk mengambil atau mengurai respons pernyataan SAMP karena kumpulan pengguna secara langsung menerima respons SAMP dari penyedia identitas Anda melalui agen pengguna.

Untuk mengonfigurasi penyedia identitas SAML 2.0 di kolam pengguna Anda


1. Masuk ke [Konsol Amazon Cognito](#). Jika diminta, masukkan AWS kredensial Anda.
2. Pilih Kolam Pengguna.
3. Pilih kolam pengguna yang ada dari daftar, atau [buat kolam pengguna](#).
4. Pilih tab Pengalaman masuk. Cari Masuk Federasi dan pilih Tambahkan penyedia identitas.
5. Pilih penyedia identitas sosial SAMP.

- Masukkan Identifier dipisahkan dengan koma. Pengidentifikasi memberi tahu Amazon Cognito bahwa ia harus memeriksa alamat email yang dimasukkan pengguna saat mereka masuk. Kemudian mengarahkan mereka ke penyedia yang sesuai dengan domain mereka.
- Pilih Tambahkan alur keluar jika Anda ingin Amazon Cognito mengirim permintaan keluar yang ditandatangani ke penyedia Anda saat pengguna keluar. Anda harus mengonfigurasi penyedia identitas SAMP 2.0 Anda untuk mengirim respons keluar ke `https://<your Amazon Cognito domain>/saml2/logout` titik akhir yang dibuat saat Anda mengonfigurasi UI yang dihosting. `saml2/logout` Titik akhir menggunakan pengikatan POST.

 Note

Jika opsi ini dipilih dan penyedia identitas SAMP Anda mengharapkan permintaan logout yang ditandatangani, Anda juga perlu mengonfigurasi sertifikat penandatanganan yang disediakan oleh Amazon Cognito dengan SAMP IDP Anda. SAMP IDP akan memproses permintaan logout yang ditandatangani dan akan mengeluarkan pengguna Anda dari sesi Amazon Cognito.

- Pilih sumber dokumen Metadata. Jika penyedia identitas Anda menawarkan metadata SAM di URL publik, Anda dapat memilih URL dokumen Metadata dan memasukkan URL publik tersebut. Jika tidak, pilih Unggah dokumen metadata dan pilih file metadata yang Anda unduh dari penyedia Anda sebelumnya.

 Note

Kami menyarankan Anda memasukkan URL dokumen metadata jika penyedia Anda memiliki titik akhir publik, daripada mengunggah file. Ini memungkinkan Amazon Cognito menyegarkan metadata secara otomatis. Biasanya, penyegaran metadata terjadi setiap 6 jam atau sebelum metadata kedaluwarsa, mana yang lebih awal.

- Pilih atribut Peta antara penyedia SAFL dan aplikasi Anda untuk memetakan atribut penyedia SAFL ke profil pengguna di kumpulan pengguna Anda. Sertakan atribut yang diperlukan kumpulan pengguna Anda di peta atribut Anda.

Misalnya, ketika Anda memilih atribut `User poolemail`, masukkan nama atribut SAMP seperti yang muncul dalam pernyataan SAMP dari penyedia identitas Anda. Penyedia identitas Anda mungkin menawarkan contoh pernyataan SAFL untuk referensi. Beberapa penyedia identitas

menggunakan nama sederhana, seperti `email`, sementara yang lain menggunakan nama atribut berformat URL, seperti contoh berikut:

```
http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress
```

## 10. Pilih Buat.

# Memulai dengan kumpulan identitas Amazon Cognito

Dengan kumpulan identitas Amazon Cognito, Anda dapat membuat identitas unik dan menetapkan izin untuk pengguna. Pangkalan identitas Anda dapat mencakup:

- Pengguna di pangkalan pengguna Amazon Cognito
- Pengguna yang melakukan autentikasi dengan penyedia identitas eksternal seperti Facebook, Google, Apple, atau penyedia identitas OIDC atau SAMP.
- Pengguna yang diautentikasi melalui proses autentikasi yang selama ini Anda gunakan

Dengan kumpulan identitas, Anda dapat memperoleh AWS kredensi sementara dengan izin yang Anda tetapkan untuk mengakses sumber daya lain secara langsung Layanan AWS atau mengakses sumber daya melalui Amazon API Gateway.

Topik

- [Buat kumpulan identitas di Amazon Cognito](#)
- [Siapkan SDK](#)
- [Integrasikan penyedia identitas](#)
- [Dapatkan kredensial](#)

## Buat kumpulan identitas di Amazon Cognito

Anda dapat membuat kumpulan identitas melalui konsol Amazon Cognito, atau Anda dapat menggunakan ( AWS Command Line Interface CLI) atau Amazon Cognito API.

Untuk membuat kolam identitas baru di konsol

1. Masuk ke [konsol Amazon Cognito](#) dan pilih Identity pool.
2. Pilih Buat kumpulan identitas.
3. Di Konfigurasi kepercayaan kumpulan identitas, pilih untuk menyiapkan kumpulan identitas Anda untuk akses Terotentikasi, akses Tamu, atau keduanya.
  - Jika Anda memilih Akses yang diautentikasi, pilih satu atau beberapa jenis Identitas yang ingin Anda tetapkan sebagai sumber identitas yang diautentikasi di kumpulan identitas

Anda. Jika mengonfigurasi penyedia pengembang Kustom, Anda tidak dapat memodifikasi atau menghapusnya setelah membuat kumpulan identitas.

4. Di Konfigurasi izin, pilih peran IAM default untuk pengguna yang diautentikasi atau tamu di kumpulan identitas Anda.
  - a. Pilih untuk Membuat peran IAM baru jika Anda ingin Amazon Cognito membuat peran baru untuk Anda dengan izin dasar dan hubungan kepercayaan dengan kumpulan identitas Anda. Masukkan nama peran IAM untuk mengidentifikasi peran baru Anda, misalnya `myidentitypool_authenticatedrole`. Pilih Lihat dokumen kebijakan untuk meninjau izin yang akan ditetapkan Amazon Cognito ke peran IAM baru Anda.
  - b. Anda dapat memilih untuk Menggunakan peran IAM yang ada jika Anda sudah memiliki peran Akun AWS yang ingin Anda gunakan. Anda harus mengonfigurasi kebijakan kepercayaan peran IAM Anda untuk disertakan `cognito-identity.amazonaws.com`. Konfigurasi kebijakan kepercayaan peran Anda agar hanya mengizinkan Amazon Cognito mengambil peran saat menampilkan bukti bahwa permintaan tersebut berasal dari pengguna yang diautentikasi di kumpulan identitas spesifik Anda. Untuk informasi selengkapnya, lihat [Kepercayaan peran dan izin](#).
5. Di Connect identity providers, masukkan detail penyedia identitas (IdPs) yang Anda pilih di Configure identity pool trust. Anda mungkin diminta untuk memberikan informasi klien aplikasi OAuth, memilih kumpulan pengguna Amazon Cognito, memilih IDP IAM, atau memasukkan pengenal khusus untuk penyedia pengembang.
  - a. Pilih pengaturan Peran untuk setiap IDP. Anda dapat menetapkan pengguna dari IDP tersebut peran Default yang Anda atur saat mengonfigurasi peran Terautentikasi, atau Anda dapat Memilih peran dengan aturan. Dengan IdP kumpulan pengguna Amazon Cognito, Anda juga dapat Memilih peran dengan `preferred_role` dalam token. Untuk informasi lebih lanjut tentang `cognito:preferred_role` klaim, lihat [Menetapkan nilai prioritas ke grup](#).
    - i. Jika Anda memilih Pilih peran dengan aturan, masukkan Klaim sumber dari autentikasi pengguna Anda, Operator yang ingin Anda bandingkan dengan klaim, Nilai yang akan menyebabkan kecocokan dengan pilihan peran ini, dan Peran yang ingin Anda tetapkan saat penetapan Peran cocok. Pilih Tambahkan yang lain untuk membuat aturan tambahan berdasarkan kondisi yang berbeda.
    - ii. Pilih resolusi Peran. Jika klaim pengguna tidak sesuai dengan aturan, Anda dapat menolak kredensial atau mengeluarkan kredensi untuk peran Terautentikasi Anda.

- b. Konfigurasi atribut untuk kontrol akses untuk setiap IDP. Atribut untuk kontrol akses memetakan klaim pengguna ke [tag utama](#) yang diterapkan Amazon Cognito untuk sesi sementara mereka. Anda dapat membuat kebijakan IAM untuk memfilter akses pengguna berdasarkan tag yang Anda terapkan pada sesi mereka.
  - i. Untuk tidak menerapkan tag utama, pilih Tidak aktif.
  - ii. Untuk menerapkan tag utama berdasarkan sub dan aud klaim, pilih Gunakan pemetaan default.
  - iii. Untuk membuat skema atribut kustom Anda sendiri ke tag utama, pilih Gunakan pemetaan khusus. Kemudian masukkan kunci Tag yang ingin Anda sumber dari setiap Klaim yang ingin Anda wakili dalam tag.
6. Di Konfigurasi properti, masukkan Nama di bawah nama kumpulan Identitas.
7. Di bawah Autentikasi dasar (klasik), pilih apakah Anda ingin Aktifkan aliran dasar. Dengan aliran dasar aktif, Anda dapat melewati pilihan peran yang dibuat untuk Anda IdPs dan menelepon [AssumeRoleWithWebIdentity](#) secara langsung. Untuk informasi selengkapnya, lihat [Aliran otentikasi kumpulan identitas \(identitas federasi\)](#).
8. Di bawah Tag, pilih Tambahkan tag jika Anda ingin menerapkan [tag](#) ke kumpulan identitas Anda.
9. Di Tinjau dan buat, konfirmasi pilihan yang Anda buat untuk kumpulan identitas baru Anda. Pilih Edit untuk kembali ke wizard dan mengubah pengaturan apa pun. Setelah selesai, pilih Buat kumpulan identitas.

## Siapkan SDK

Untuk menggunakan kumpulan identitas Amazon Cognito, siapkan, file AWS Amplify AWS SDK for Java, atau file. AWS SDK for .NET Untuk informasi selengkapnya, lihat topik berikut.

- [Menyiapkan SDK untuk JavaScript](#) di Panduan AWS SDK for Java Pengembang
- [Amplify Dokumentasi](#) di Amplify Dev Center
- [Penyedia kredensi Amazon Cognito](#) di Panduan Pengembang AWS SDK for .NET

## Integrasikan penyedia identitas

Kumpulan identitas Amazon Cognito (identitas federasi) mendukung otentikasi pengguna melalui kumpulan pengguna Amazon Cognito, penyedia identitas federasi—termasuk penyedia identitas Amazon, Facebook, Google, Apple, dan SAML—dan identitas yang tidak diautentikasi. Fitur ini juga

mendukung [Identitas yang diautentikasi pengembang \(kumpulan identitas\)](#), yang memungkinkan Anda mendaftar dan mengautentikasi pengguna melalui proses otentikasi backend Anda sendiri.

Untuk belajar lebih lanjut tentang menggunakan kolam pengguna Amazon Cognito dalam membuat direktori pengguna Anda sendiri, lihat [Kolam pengguna Amazon Cognito](#) dan [Mengakses Layanan AWS menggunakan kumpulan identitas setelah masuk](#).

Untuk belajar selengkapnya tentang penggunaan penyedia identitas eksternal, lihat [Identitas mengumpulkan penyedia identitas eksternal](#).

Untuk mempelajari lebih lanjut tentang mengintegrasikan proses otentikasi backend Anda sendiri, lihat [Identitas yang diautentikasi pengembang \(kumpulan identitas\)](#).

## Dapatkan kredensial

Kumpulan identitas Amazon Cognito menyediakan AWS kredensi sementara untuk pengguna yang merupakan tamu (tidak diautentikasi) dan untuk pengguna yang telah mengautentikasi dan menerima token. Dengan AWS kredensialnya, aplikasi Anda dapat mengakses backend dengan aman di dalam AWS atau di luar melalui Amazon API AWS Gateway. Lihat [Mendapatkan kredensial](#).



# Opsi penyiapan terpandu untuk Amazon Cognito

Anda mungkin ingin mengevaluasi fitur Amazon Cognito dalam pengalaman terstruktur dan terpandu. Berikut adalah beberapa sumber daya eksternal yang memberikan pengalaman yang disesuaikan dengan kumpulan pengguna dan kumpulan identitas.

## Selesaikan lokakarya

AWS studio lokakarya [menyelenggarakan lokakarya](#) yang memandu Anda melalui pengaturan sebagian besar fitur Amazon Cognito. Fitur-fitur ini termasuk API kumpulan pengguna, UI yang dihosting kumpulan pengguna, kumpulan identitas, dan konfigurasi keamanan.

## Tambahkan kode aplikasi dari contoh

Bab [contoh kode](#) dalam panduan ini memiliki kode aplikasi yang dapat Anda gunakan dengan kumpulan pengguna dan kumpulan identitas. Bagian kumpulan pengguna dari chapter contoh kode memiliki cuplikan pendek yang mencakup operasi individual, dan contoh yang lebih panjang end-to-end misalnya aplikasi dalam berbagai bahasa pemrograman.

## Buat aplikasi fullstack dengan AWS Amplify

[AWS Amplify](#) adalah Layanan AWS untuk pengembang yang ingin mengembangkan dan meng-host aplikasi dan antarmuka pengguna. Amazon Cognito adalah komponen otentikasi Amplify. Saat menambahkan autentikasi ke aplikasi, Amplify dapat mengotomatiskan penerapan kumpulan pengguna Amazon Cognito dan sumber daya kumpulan identitas. Lihat juga [Mengintegrasikan otentikasi dan otorisasi Amazon Cognito dengan aplikasi web dan seluler](#).

## Lebih banyak sumber daya aplikasi Amazon Cognito GitHub

- [Contoh alur otentikasi dengan.NET untuk Amazon Cognito](#)
- [Auth Tanpa Kata Sandi Amazon Cognito](#)
- [PetStorecontoh dengan Izin Terverifikasi Amazon](#)
- [Contoh Aplikasi React Menggunakan ABAC+Identity Pools untuk Mengakses Sumber Daya AWS](#)
- [Mesin berbasis Amazon Cognito dan API Gateway untuk otorisasi mesin menggunakan CDK AWS](#)
- [Membangun otorisasi berbutir halus menggunakan Amazon Cognito, API Gateway, dan IAM](#)
- [CloudFrontotorisasi @edge](#)

## Lebih banyak lokakarya

- [Menerapkan otentikasi Tanpa Kata Sandi dengan Amazon Cognito dan WebAuthn](#)
- [Identitas SaaS multi-penyewa dengan kumpulan pengguna Amazon Cognito](#)
- [Amazon Cognito JWT Deep Dive](#)

# Mengintegrasikan otentikasi dan otorisasi Amazon Cognito dengan aplikasi web dan seluler

Saat mengintegrasikan aplikasi dengan klien aplikasi Amazon Cognito, Anda dapat menjalankan operasi API untuk autentikasi dan otorisasi pengguna. Kami menyarankan Anda menggunakan [AWS Amplify](#) untuk mengintegrasikan Amazon Cognito dengan web dan aplikasi seluler Anda. AWS Amplify adalah solusi lengkap yang memungkinkan pengembang web dan seluler frontend dengan mudah membangun, menghubungkan, dan meng-host aplikasi fullstack AWS, dengan fleksibilitas untuk memanfaatkan luasnya Layanan AWS saat kasus penggunaan Anda berkembang. Amplify Auth terutama menggunakan Amazon Cognito untuk membangun fitur otentikasi.

## Topik

- [Otentikasi dengan AWS Amplify](#)
- [Otentikasi dengan AWS SDK](#)
- [Otorisasi dengan Izin Terverifikasi Amazon](#)

Implementasi khas Amazon Cognito menggunakan campuran alat visual dan API. Konsol Amazon Cognito adalah antarmuka visual untuk penyiapan dan pengelolaan kumpulan pengguna Amazon Cognito dan kumpulan identitas Anda. UI yang dihosting adalah aplikasi masuk ready-to-use berbasis web untuk pengujian cepat dan penyebaran kumpulan pengguna Amazon Cognito. Selain itu, di sebagian besar penerapan Amazon Cognito, Anda harus menambahkan kode di aplikasi untuk berinteraksi dengan kumpulan pengguna dan kumpulan identitas. Misalnya, aplikasi Anda mungkin memanggil UI yang dihosting untuk login pengguna, lalu memanggil titik akhir token dari kode aplikasi Anda untuk menukar kode otorisasi pengguna Anda dengan token. Kemudian aplikasi Anda harus menafsirkan dan menyimpan token pengguna Anda, dan menyajikannya dalam konteks yang sesuai untuk otentikasi dan otorisasi. Amplify menambahkan alat integrasi terpandu dengan fungsi bawaan untuk proses ini.

Anda juga dapat membangun sumber daya Amazon Cognito sepenuhnya dalam kode. [Untuk memulai dengan kode aplikasi kustom Anda sendiri, kunjungi contoh kode Amazon Cognito untuk AWS SDK. Untuk integrasi dengan Amazon Cognito sebagai penyedia identitas OpenID Connect, gunakan alat pengembang OpenID Connect.](#)

Sebelum Anda menggunakan otentikasi dan otorisasi Amazon Cognito, pilih platform aplikasi dan siapkan kode Anda untuk diintegrasikan dengan layanan. Untuk platform yang tersedia,

lihat [Otentikasi dengan AWS SDK](#). AWS CLI Ini adalah SDK baris perintah untuk Amazon Cognito dan lainnya Layanan AWS, dan merupakan tempat yang berharga untuk mulai membiasakan diri dengan Amazon Cognito API.

#### Note

Beberapa komponen Amazon Cognito hanya dapat dikonfigurasi dengan API. Misalnya, Anda hanya dapat menyetel pemicu Lambda [SMS atau pengirim email kustom](#) kumpulan pengguna dengan permintaan yang memperbarui LambdaConfig properti kelas dalam permintaan [UpdateUserPool](#) atau [CreateUserPool](#) API. [UserPool](#)

API kumpulan pengguna Amazon Cognito membagikan namespace-nya dengan beberapa kelas operasi API. Satu kelas mengonfigurasi kumpulan pengguna dan prosesnya, penyedia identitas, dan pengguna. Lain termasuk operasi yang tidak diautentikasi untuk pengguna Anda di klien publik untuk masuk, keluar, dan mengelola profil mereka. Kelas terakhir operasi API melakukan operasi pengguna yang Anda otorisasi dengan AWS kredensialnya sendiri di klien sisi server rahasia. Anda harus mengetahui arsitektur aplikasi yang Anda inginkan sebelum mulai menerapkan kode aplikasi. Untuk informasi selengkapnya, lihat [Menggunakan API kumpulan pengguna Amazon Cognito dan titik akhir kumpulan pengguna](#).

## Otentikasi dengan AWS Amplify

AWS Amplify adalah solusi lengkap untuk membangun aplikasi web dan seluler. Dengan Amplify, Anda dapat terhubung ke sumber daya yang ada dengan pustaka Amplify, atau Anda dapat membuat dan mengonfigurasi sumber daya baru dengan antarmuka baris perintah Amplify (CLI). Amplify juga memiliki komponen UI yang terhubung seperti [Authenticator](#) untuk penyiapan dan penyesuaian pengalaman masuk dan pendaftaran di aplikasi Anda.

Untuk menggunakan fitur otentikasi Amplify di aplikasi front-end Anda, lihat dokumentasi berikut berdasarkan platform.

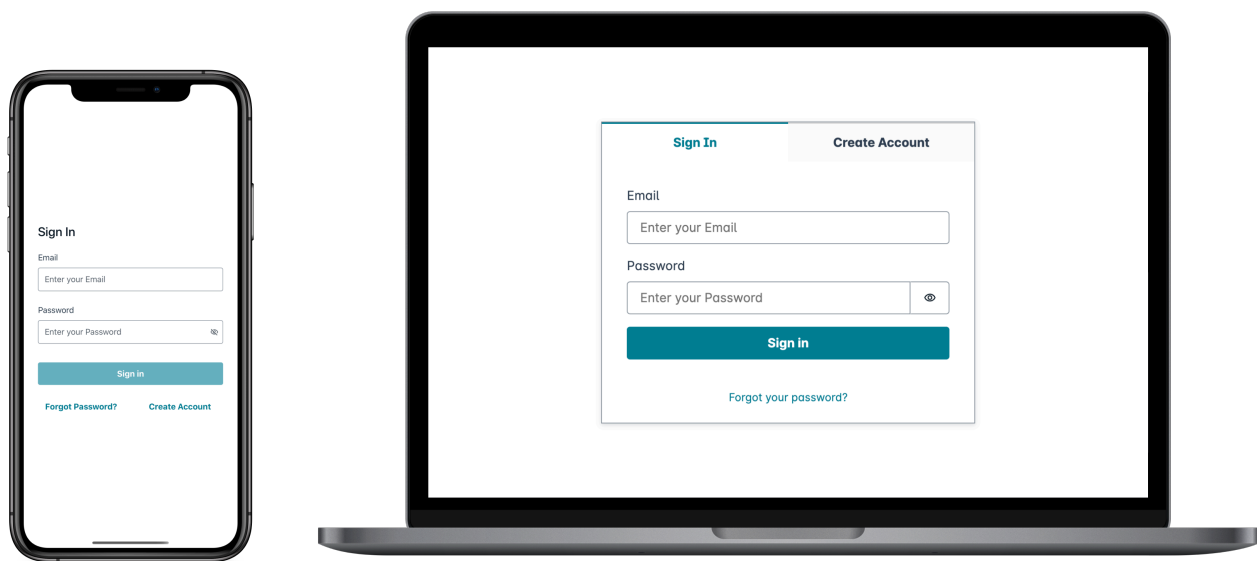
- [Amplify otentikasi untuk JavaScript](#)
- [Amplify otentikasi untuk iOS](#)
- [Amplify otentikasi untuk Android](#)
- [Amplify otentikasi untuk Flutter](#)

Library Amplify adalah open source dan tersedia di [GitHub](#) Untuk mempelajari selengkapnya tentang cara Amplify Auth mengimplementasikan autentikasi Amazon Cognito, kunjungi library berikut.

- [amplify-js](#)
- [amplify-cepat](#)
- [amplify-flutter](#)
- [amplify-android](#)

## Membuat antarmuka pengguna (UI) dengan Amplify

[UI yang dihosting kumpulan pengguna Amazon Cognito](#) dapat memenuhi kebutuhan penting dari front-end otentikasi untuk aplikasi web atau seluler. Untuk menyesuaikan antarmuka pengguna (UI) Anda di luar parameter yang diakomodasi oleh UI yang dihosting, buat aplikasi khusus. [Amplify UI](#) adalah kumpulan komponen front-end yang dapat disesuaikan dalam berbagai bahasa.



Untuk memulai dengan komponen otentikasi kustom Anda, kunjungi dokumentasi berikut untuk komponen Authenticator.

- [Authenticator untuk Android](#)
- [Authenticator untuk Angular](#)
- [Authenticator untuk Flutter](#)

- [Authenticator untuk React](#)
- [Authenticator untuk React Native](#)
- [Authenticator untuk Swift](#)
- [Authenticator untuk Vue](#)

## Otentikasi dengan AWS SDK

Untuk menggunakan backend aman untuk membangun layanan mikro identitas Anda sendiri yang berinteraksi dengan Amazon Cognito, sambungkan ke kumpulan pengguna Amazon Cognito dan API kumpulan identitas Amazon Cognito dengan SDK dalam bahasa pilihan Anda. AWS

Untuk detail tentang setiap operasi API, lihat Referensi API [kumpulan pengguna Amazon Cognito](#) dan [Referensi API Amazon Cognito](#). Dokumen-dokumen ini berisi [Lihat juga](#) bagian dengan sumber daya untuk menggunakan berbagai SDK di platform yang didukung.

- [AWS Antarmuka Baris Perintah](#)
- [AWS SDK for .NET](#)
- [AWS SDK for C++](#)
- [AWS SDK for Go](#)
- [AWS SDK for Java V2](#)
- [AWS SDK untuk JavaScript](#)
- [AWS SDK for PHP V3](#)
- [AWS SDK untuk Python](#)
- [AWS SDK for Ruby V3](#)

## Otorisasi dengan Izin Terverifikasi Amazon

[Izin Terverifikasi Amazon](#) adalah layanan otorisasi untuk aplikasi yang Anda buat. Saat Anda menambahkan kumpulan pengguna Amazon Cognito sebagai sumber identitas, aplikasi Anda dapat meneruskan akses kumpulan pengguna atau token identitas (ID) ke Izin Terverifikasi untuk mengizinkan atau menolak keputusan. Izin Terverifikasi mempertimbangkan properti pengguna Anda dan konteks permintaan berdasarkan kebijakan yang Anda tulis dalam Bahasa [Kebijakan Cedar](#). Konteks permintaan dapat menyertakan pengenalan untuk dokumen, gambar, atau sumber daya lain yang mereka minta, dan tindakan yang ingin diambil pengguna Anda pada sumber daya tersebut.

Aplikasi Anda dapat memberikan identitas pengguna atau token akses ke Izin Terverifikasi [IsAuthorizedWithToken](#) atau permintaan [BatchIsAuthorizedWithToken](#) API. Operasi API ini menerima pengguna Anda sebagai Principal dan membuat keputusan otorisasi untuk Resource hal yang ingin mereka akses. Action Kustom tambahan Context dapat berkontribusi pada keputusan akses terperinci.

Saat aplikasi Anda menampilkan token dalam permintaan IsAuthorizedWithToken API, Izin Terverifikasi akan melakukan validasi berikut.

1. Kumpulan pengguna Anda adalah [sumber identitas Izin](#) Terverifikasi yang dikonfigurasi untuk penyimpanan kebijakan yang diminta.
2. audKlaim client\_id atau, masing-masing dalam token akses atau identitas Anda, cocok dengan ID klien aplikasi kumpulan pengguna yang Anda berikan ke Izin Terverifikasi. Untuk memverifikasi klaim ini, Anda harus [mengonfigurasi validasi ID klien di sumber](#) identitas Izin Terverifikasi.
3. Token Anda tidak kedaluwarsa.
4. Nilai token\_use klaim dalam token Anda cocok dengan parameter yang Anda kirimkan IsAuthorizedWithToken. token\_useKlaim harus access jika Anda meneruskannya ke accessToken parameter, dan id jika Anda meneruskannya ke identityToken parameter.
5. Tanda tangan di token Anda berasal dari kunci web JSON yang diterbitkan (JWK) dari kumpulan pengguna Anda. Anda dapat melihat JWK Anda di [https://cognito-idp.\*Region\*.amazonaws.com/\*your user pool ID\*/.well-known/jwks.json](https://cognito-idp.<i>Region</i>.amazonaws.com/<i>your user pool ID</i>/.well-known/jwks.json)

### Token yang dicabut dan pengguna yang dihapus

Izin Terverifikasi hanya memvalidasi informasi yang diketahui dari sumber identitas Anda dan dari waktu kedaluwarsa token pengguna Anda. Izin Terverifikasi tidak memeriksa pencabutan token atau keberadaan pengguna. Jika Anda mencabut token pengguna atau menghapus profil pengguna dari kumpulan pengguna, Izin Terverifikasi masih menganggap token tersebut valid hingga kedaluwarsa.

### Evaluasi kebijakan

Konfigurasi kumpulan pengguna Anda sebagai [sumber identitas](#) untuk [toko kebijakan](#) Anda. Konfigurasi aplikasi Anda untuk mengirimkan token pengguna Anda dalam permintaan ke Izin Terverifikasi. Untuk setiap permintaan, Izin Terverifikasi membandingkan klaim dalam token dengan kebijakan. Kebijakan Izin Terverifikasi seperti kebijakan IAM di AWS Ini menyatakan prinsip, sumber daya, dan tindakan. Izin Terverifikasi merespons permintaan Anda Allow jika cocok dengan tindakan yang diizinkan dan tidak cocok dengan tindakan eksplisit; jika tidak, ia Deny merespons dengan.

Deny Untuk informasi selengkapnya, lihat [kebijakan Izin Terverifikasi Amazon](#) di Panduan Pengguna Izin Terverifikasi Amazon.

## Menyesuaikan token

Untuk mengubah, menambah, dan menghapus klaim pengguna yang ingin Anda tampilkan ke Izin Terverifikasi, sesuaikan konten dalam token akses dan identitas Anda dengan [Pemicu Lambda generasi pra token](#) file. Dengan pemicu pembuatan token pra, Anda dapat menambahkan dan memodifikasi klaim di token Anda. Misalnya, Anda dapat menanyakan database untuk atribut pengguna tambahan dan menyandikannya ke dalam token ID Anda.

### Note

Karena cara Izin Terverifikasi memproses klaim, jangan tambahkan klaim bernama `cognitodev`, atau `custom` dalam fungsi pembuatan token pra Anda. Ketika Anda menunjukkan awalan klaim yang dicadangkan ini tidak dalam format yang dibatasi titik dua seperti `cognito:username` tetapi sebagai nama klaim lengkap, permintaan otorisasi Anda gagal.

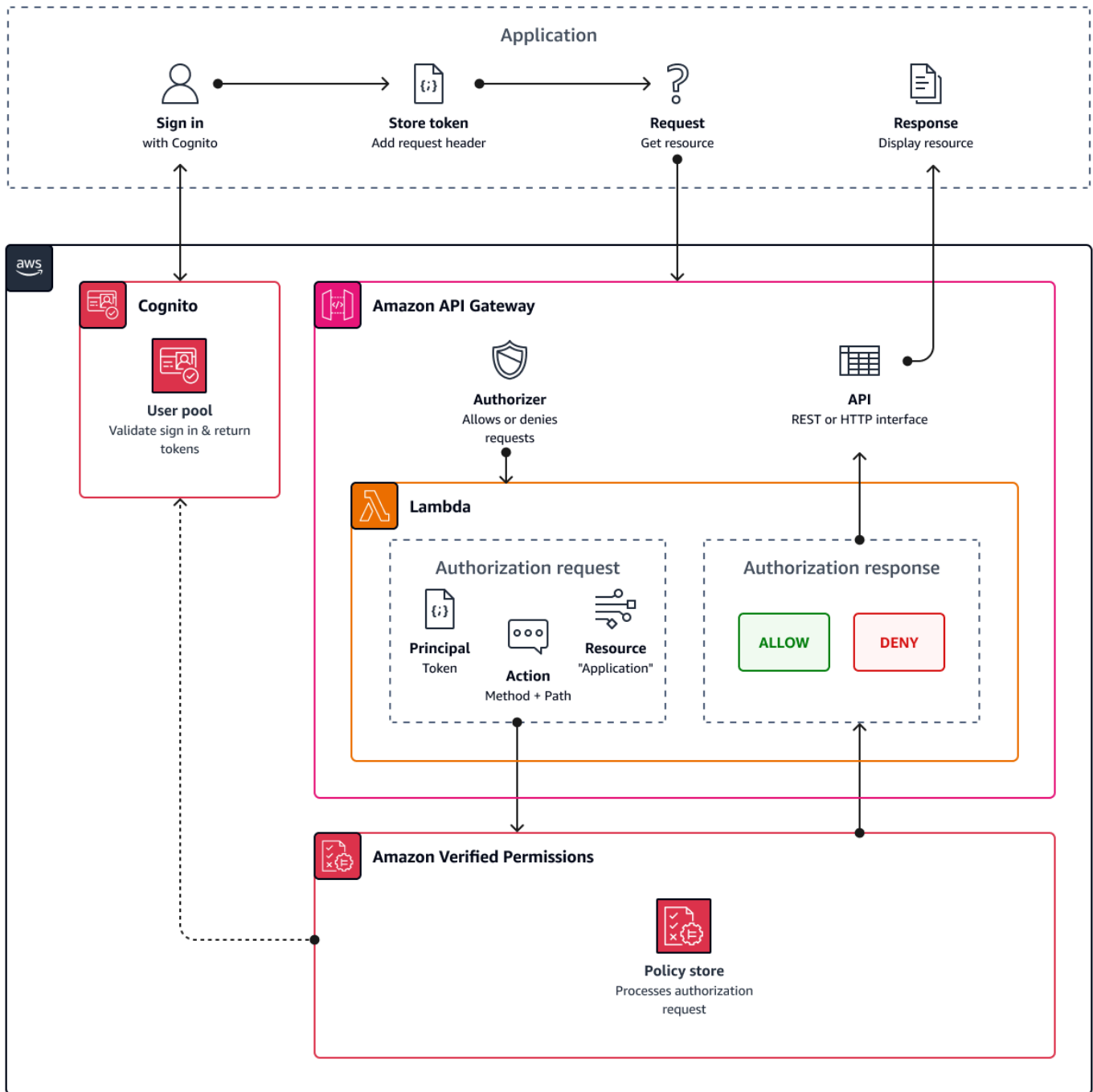
Untuk informasi selengkapnya tentang cara Izin Terverifikasi memetakan klaim di token Amazon Cognito ke kebijakan otorisasi, lihat [Memetakan token Amazon Cognito](#) ke skema Izin Terverifikasi.

## Otorisasi API dengan Izin Terverifikasi

ID atau token akses Anda dapat mengotorisasi permintaan ke back-end Amazon API Gateway REST API dengan Izin Terverifikasi. Anda dapat membuat [toko kebijakan](#) dengan tautan langsung ke kumpulan pengguna dan API Anda. Dengan opsi mulai [Mengatur dengan Cognito dan API Gateway](#), Izin Terverifikasi menambahkan sumber identitas kumpulan pengguna ke penyimpanan kebijakan, dan otorisasi Lambda ke API. Saat aplikasi Anda meneruskan token pembawa kumpulan pengguna ke API, otorisasi Lambda akan memanggil Izin Terverifikasi. Authorizer meneruskan token sebagai prinsipal dan jalur permintaan serta metode sebagai tindakan.

Diagram berikut menggambarkan alur otorisasi untuk API Gateway API dengan Izin Terverifikasi. Untuk perincian terperinci, lihat [penyimpanan kebijakan terkait API](#) di Panduan Pengguna Izin Terverifikasi Amazon.





Izin Terverifikasi menyusun otorisasi API di sekitar grup [kumpulan pengguna](#). Karena ID dan token akses menyertakan `cognito:groups` klaim, toko kebijakan Anda dapat mengelola kontrol akses berbasis peran (RBAC) untuk API Anda dalam berbagai konteks aplikasi.

## Memilih pengaturan toko kebijakan

Saat mengonfigurasi sumber identitas di toko kebijakan, Anda harus memilih apakah Anda ingin memproses akses atau token ID. Keputusan ini penting bagi cara mesin kebijakan Anda beroperasi. Token ID berisi atribut pengguna. Token akses berisi informasi kontrol akses pengguna: cakupan [OAuth](#). Meskipun kedua jenis token memiliki informasi keanggotaan grup, kami umumnya merekomendasikan token akses untuk RBAC dengan penyimpanan kebijakan Izin Terverifikasi. Token akses menambah keanggotaan grup dengan cakupan yang dapat berkontribusi pada keputusan otorisasi. Klaim dalam token akses menjadi [konteks](#) dalam permintaan otorisasi.

Anda juga harus mengonfigurasi tipe entitas pengguna dan grup saat mengonfigurasi kumpulan pengguna sebagai sumber identitas. Jenis entitas adalah pengidentifikasi utama, tindakan, dan sumber daya yang dapat Anda referensikan dalam kebijakan Izin Terverifikasi. Entitas di toko kebijakan dapat memiliki hubungan keanggotaan, di mana satu entitas dapat menjadi anggota entitas induk. Dengan keanggotaan, Anda dapat mereferensikan grup utama, grup tindakan, dan grup sumber daya. Dalam kasus grup kumpulan pengguna, jenis entitas pengguna yang Anda tentukan harus merupakan anggota dari jenis entitas grup. Saat menyiapkan [penyimpanan kebijakan terkait API](#) atau mengikuti Penyiapan terpandu di konsol Izin Terverifikasi, toko kebijakan Anda secara otomatis memiliki hubungan orang tua-anggota ini.

Token ID dapat menggabungkan RBAC dengan kontrol akses berbasis atribut (ABAC). Setelah membuat [toko kebijakan terkait API, Anda dapat menyempurnakan kebijakan](#) Anda dengan [atribut pengguna](#) dan keanggotaan grup. Klaim atribut dalam token ID menjadi [atribut utama](#) dalam permintaan otorisasi. Kebijakan Anda dapat membuat keputusan otorisasi berdasarkan atribut utama.

Anda juga dapat mengonfigurasi toko kebijakan untuk menerima token dengan `client_id` klaim `aud` atau yang cocok dengan daftar klien aplikasi yang dapat diterima yang Anda berikan.

## Contoh kebijakan untuk otorisasi API berbasis peran

Contoh kebijakan berikut dibuat dengan penyiapan penyimpanan kebijakan Izin Terverifikasi untuk [PetStore](#) contoh REST API.

```
permit(  
  principal in PetStore::UserGroup::"us-east-1_EXAMPLE|MyGroup",  
  action in [ PetStore::Action::"get /pets", PetStore::Action::"get /pets/{petId}" ],  
  resource  
);
```

Izin Terverifikasi mengembalikan Allow keputusan untuk permintaan otorisasi dari aplikasi Anda ketika:

1. Aplikasi Anda meneruskan ID atau token akses di Authorization header sebagai token pembawa.
2. Aplikasi Anda melewati token dengan `cognito:groups` klaim yang berisi `stringMyGroup`.
3. Aplikasi Anda membuat HTTP GET permintaan untuk, misalnya, `https://myapi.example.com/pets` atau `https://myapi.example.com/pets/scrappy`.

## Contoh kebijakan untuk pengguna Amazon Cognito

Kumpulan pengguna Anda juga dapat menghasilkan permintaan otorisasi ke Izin Terverifikasi dalam kondisi selain permintaan API. Anda dapat mengirimkan keputusan kontrol akses apa pun dalam aplikasi Anda ke toko kebijakan Anda. Misalnya, Anda dapat melengkapi keamanan Amazon DynamoDB atau Amazon S3 dengan kontrol akses berbasis atribut sebelum permintaan apa pun transit ke jaringan, sehingga mengurangi penggunaan kuota.

Contoh berikut menggunakan [Bahasa Kebijakan Cedar](#) untuk mengizinkan pengguna Keuangan yang melakukan autentikasi dengan satu klien aplikasi kumpulan pengguna untuk membaca dan menulis. `example_image.png` John, pengguna di aplikasi Anda, menerima token ID dari klien aplikasi Anda dan meneruskannya dalam permintaan GET ke URL yang memerlukan otorisasi. `https://example.com/images/example_image.png` Token ID John memiliki `aud` klaim ID klien aplikasi kumpulan pengguna `Anda1234567890example`. Fungsi Lambda generasi pra token Anda juga memasukkan klaim baru `costCenter` dengan nilai, untuk John, dari `Finance1234`

```

permit (
  principal,
  actions in [ExampleCorp::Action::"readFile", "writeFile"],
  resource == ExampleCorp::Photo::"example_image.png"
)
when {
  principal.aud == "1234567890example" &&
  principal.custom.costCenter like "Finance*"
};

```

Badan permintaan berikut menghasilkan Allow respons.

```
{
```

```

"accesstoken": "[John's ID token]",
"action": {
  "actionId": "readFile",
  "actionType": "Action"
},
"resource": {
  "entityId": "example_image.png",
  "entityType": "Photo"
}
}

```

Bila Anda ingin menentukan prinsipal dalam kebijakan Izin Terverifikasi, gunakan format berikut:

```

permit (
  principal == [Namespace]::[Entity]::"[user pool ID]"|"[user sub]",
  action,
  resource
);

```

Berikut ini adalah contoh utama untuk pengguna dalam kumpulan pengguna dengan ID `us-east-1_Example` dengan sub, atau ID pengguna, `973db890-092c-49e4-a9d0-912a4c0a20c7`.

```

principal == ExampleCorp::User::"us-east-1_Example|973db890-092c-49e4-a9d0-912a4c0a20c7",

```

Bila Anda ingin menentukan grup pengguna dalam kebijakan Izin Terverifikasi, gunakan format berikut:

```

permit (
  principal in [Namespace]::[Group Entity]::"[Group name]",
  action,
  resource
);

```

Berikut ini adalah contoh

### Kontrol akses berbasis atribut

Otorisasi dengan Izin Terverifikasi untuk aplikasi Anda, dan [atribut untuk fitur kontrol akses](#) kumpulan identitas Amazon Cognito AWS untuk kredensial, keduanya merupakan bentuk kontrol akses berbasis atribut (ABAC). Berikut ini adalah perbandingan fitur Izin Terverifikasi dan Amazon Cognito ABAC.

Di ABAC, sistem memeriksa atribut entitas dan membuat keputusan otorisasi dari kondisi yang Anda tentukan.

Layanan	Proses	Hasil
Izin Terverifikasi Amazon	Mengembalikan Allow atau Deny keputusan dari analisis kumpulan pengguna JWT.	Akses ke sumber daya aplikasi berhasil atau gagal berdasarkan evaluasi kebijakan Cedar.
Kumpulan identitas Amazon Cognito (atribut untuk kontrol akses)	Menetapkan <a href="#">tag sesi</a> ke pengguna Anda berdasarkan atributnya. Ketentuan kebijakan IAM dapat memeriksa tag Allow atau akses Deny pengguna ke Layanan AWS.	Sesi yang ditandai dengan AWS kredensi sementara untuk peran IAM.

# Contoh kode untuk Amazon Cognito menggunakan AWS SDK

Contoh kode berikut ini menunjukkan cara menggunakan Amazon Cognito dengan AWS perangkat pengembangan perangkat lunak (SDK).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Contoh kode

- [Contoh kode untuk Identitas Amazon Cognito menggunakan SDK AWS](#)
  - [Tindakan untuk Identitas Amazon Cognito menggunakan SDK AWS](#)
    - [Gunakan CreateIdentityPool dengan AWS SDK atau CLI](#)
    - [Gunakan DeleteIdentityPool dengan AWS SDK atau CLI](#)
    - [Gunakan DescribeIdentityPool dengan AWS SDK atau CLI](#)
    - [Gunakan GetCredentialsForIdentity dengan AWS SDK atau CLI](#)
    - [Gunakan GetIdentityPoolRoles dengan AWS SDK atau CLI](#)
    - [Gunakan ListIdentityPools dengan AWS SDK atau CLI](#)
    - [Gunakan SetIdentityPoolRoles dengan AWS SDK atau CLI](#)
    - [Gunakan UpdateIdentityPool dengan AWS SDK atau CLI](#)
  - [Contoh lintas layanan untuk Identitas Amazon Cognito menggunakan SDK AWS](#)
    - [Membangun aplikasi Amazon Transcribe](#)
    - [Membuat aplikasi penjelajah Amazon Textract](#)
- [Contoh kode untuk Penyedia Identitas Amazon Cognito menggunakan SDK AWS](#)
  - [Tindakan untuk Penyedia Identitas Amazon Cognito menggunakan SDK AWS](#)
    - [Gunakan AdminCreateUser dengan AWS SDK atau CLI](#)
    - [Gunakan AdminGetUser dengan AWS SDK atau CLI](#)
    - [Gunakan AdminInitiateAuth dengan AWS SDK atau CLI](#)
    - [Gunakan AdminRespondToAuthChallenge dengan AWS SDK atau CLI](#)
    - [Gunakan AdminSetUserPassword dengan AWS SDK atau CLI](#)
    - [Gunakan AssociateSoftwareToken dengan AWS SDK atau CLI](#)

- [Gunakan ConfirmDevice dengan AWS SDK atau CLI](#)
- [Gunakan ConfirmForgotPassword dengan AWS SDK atau CLI](#)
- [Gunakan ConfirmSignUp dengan AWS SDK atau CLI](#)
- [Gunakan CreateUserPool dengan AWS SDK atau CLI](#)
- [Gunakan CreateUserPoolClient dengan AWS SDK atau CLI](#)
- [Gunakan DeleteUser dengan AWS SDK atau CLI](#)
- [Gunakan ForgotPassword dengan AWS SDK atau CLI](#)
- [Gunakan InitiateAuth dengan AWS SDK atau CLI](#)
- [Gunakan ListUserPools dengan AWS SDK atau CLI](#)
- [Gunakan ListUsers dengan AWS SDK atau CLI](#)
- [Gunakan ResendConfirmationCode dengan AWS SDK atau CLI](#)
- [Gunakan RespondToAuthChallenge dengan AWS SDK atau CLI](#)
- [Gunakan SignUp dengan AWS SDK atau CLI](#)
- [Gunakan UpdateUserPool dengan AWS SDK atau CLI](#)
- [Gunakan VerifySoftwareToken dengan AWS SDK atau CLI](#)
- [Skenario untuk Penyedia Identitas Amazon Cognito menggunakan SDK AWS](#)
  - [Secara otomatis mengonfirmasi pengguna Amazon Cognito yang dikenal dengan fungsi Lambda menggunakan SDK AWS](#)
  - [Secara otomatis memigrasikan pengguna Amazon Cognito yang dikenal dengan fungsi Lambda menggunakan SDK AWS](#)
  - [Mendaftar pengguna dengan kumpulan pengguna Amazon Cognito yang memerlukan MFA menggunakan SDK AWS](#)
  - [Menulis data aktivitas kustom dengan fungsi Lambda setelah autentikasi pengguna Amazon Cognito menggunakan SDK AWS](#)
- [Contoh kode untuk Sinkronisasi Amazon Cognito menggunakan SDK AWS](#)
  - [Tindakan untuk Sinkronisasi Amazon Cognito menggunakan SDK AWS](#)
    - [Gunakan ListIdentityPoolUsage dengan AWS SDK atau CLI](#)

# Contoh kode untuk Identitas Amazon Cognito menggunakan SDK AWS

Contoh kode berikut menunjukkan cara menggunakan Identitas Amazon Cognito dengan kit pengembangan AWS perangkat lunak (SDK).

Tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Meskipun tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks pada skenario terkait dan contoh lintas layanan.

Contoh lintas layanan adalah contoh aplikasi yang bekerja di beberapa Layanan AWS.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Contoh kode

- [Tindakan untuk Identitas Amazon Cognito menggunakan SDK AWS](#)
  - [Gunakan CreatelIdentityPool dengan AWS SDK atau CLI](#)
  - [Gunakan DeletelIdentityPool dengan AWS SDK atau CLI](#)
  - [Gunakan DescribelIdentityPool dengan AWS SDK atau CLI](#)
  - [Gunakan GetCredentialsForIdentity dengan AWS SDK atau CLI](#)
  - [Gunakan GetIdentityPoolRoles dengan AWS SDK atau CLI](#)
  - [Gunakan ListIdentityPools dengan AWS SDK atau CLI](#)
  - [Gunakan SetIdentityPoolRoles dengan AWS SDK atau CLI](#)
  - [Gunakan UpdatelIdentityPool dengan AWS SDK atau CLI](#)
- [Contoh lintas layanan untuk Identitas Amazon Cognito menggunakan SDK AWS](#)
  - [Membangun aplikasi Amazon Transcribe](#)
  - [Membuat aplikasi penjelajah Amazon Textract](#)

## Tindakan untuk Identitas Amazon Cognito menggunakan SDK AWS

Contoh kode berikut menunjukkan cara melakukan tindakan Identitas Amazon Cognito individual dengan AWS SDK. Kutipan ini memanggil Amazon Cognito Identity 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 Identitas Amazon Cognito](#).

### Contoh

- [Gunakan CreateIdentityPool dengan AWS SDK atau CLI](#)
- [Gunakan DeleteIdentityPool dengan AWS SDK atau CLI](#)
- [Gunakan DescribeIdentityPool dengan AWS SDK atau CLI](#)
- [Gunakan GetCredentialsForIdentity dengan AWS SDK atau CLI](#)
- [Gunakan GetIdentityPoolRoles dengan AWS SDK atau CLI](#)
- [Gunakan ListIdentityPools dengan AWS SDK atau CLI](#)
- [Gunakan SetIdentityPoolRoles dengan AWS SDK atau CLI](#)
- [Gunakan UpdateIdentityPool dengan AWS SDK atau CLI](#)

## Gunakan **CreateIdentityPool** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan `CreateIdentityPool`.

### CLI

#### AWS CLI

Untuk membuat kumpulan identitas dengan penyedia kumpulan identitas Cognito

Contoh ini menciptakan sebuah kumpulan identitas bernama `MyIdentityPool`. Ini memiliki penyedia kumpulan identitas Cognito. Identitas yang tidak diautentikasi tidak diperbolehkan.

Perintah:

```
aws cognito-identity create-identity-pool --identity-pool-name
  MyIdentityPool --no-allow-unauthenticated-identities --cognito-
  identity-providers ProviderName="cognito-idp.us-west-2.amazonaws.com/us-
  west-2_aaaaaaaa",ClientId="3n4b5urk1ft4f13mg5e62d9ado",ServerSideTokenCheck=false
```

Output:

```
{
```

```
"IdentityPoolId": "us-west-2:11111111-1111-1111-1111-111111111111",
"IdentityPoolName": "MyIdentityPool",
"AllowUnauthenticatedIdentities": false,
"CognitoIdentityProviders": [
  {
    "ProviderName": "cognito-idp.us-west-2.amazonaws.com/us-
west-2_1111111111",
    "ClientId": "3n4b5urk1ft4fl3mg5e62d9ado",
    "ServerSideTokenCheck": false
  }
]
```

- Untuk detail API, lihat [CreateIdentityPool](#) di Referensi AWS CLI Perintah.

## Java

### SDK untuk Java 2.x

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.cognitoidentity.CognitoIdentityClient;
import
  software.amazon.awssdk.services.cognitoidentity.model.CreateIdentityPoolRequest;
import
  software.amazon.awssdk.services.cognitoidentity.model.CreateIdentityPoolResponse;
import
  software.amazon.awssdk.services.cognitoidentityprovider.model.CognitoIdentityProviderExco

/**
 * 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 CreateIdentityPool {
    public static void main(String[] args) {
        final String usage = ""
            Usage:
                <identityPoolName>\s

            Where:
                identityPoolName - The name to give your identity pool.
            """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String identityPoolName = args[0];
        CognitoIdentityClient cognitoClient = CognitoIdentityClient.builder()
            .region(Region.US_EAST_1)
            .build();

        String identityPoolId = createIdPool(cognitoClient, identityPoolName);
        System.out.println("Unity pool ID " + identityPoolId);
        cognitoClient.close();
    }

    public static String createIdPool(CognitoIdentityClient cognitoClient, String
identityPoolName) {
        try {
            CreateIdentityPoolRequest poolRequest =
CreateIdentityPoolRequest.builder()
                .allowUnauthenticatedIdentities(false)
                .identityPoolName(identityPoolName)
                .build();

            CreateIdentityPoolResponse response =
cognitoClient.createIdentityPool(poolRequest);
            return response.identityPoolId();

        } catch (CognitoIdentityProviderException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }
}
```

```
}  
}
```

- Untuk detail API, lihat [CreateIdentityPool](#) di Referensi AWS SDK for Java 2.x API.

## PowerShell

### Alat untuk PowerShell

Contoh 1: Membuat Identity Pool baru yang memungkinkan identitas yang tidak diautentikasi.

```
New-CGIIIdentityPool -AllowUnauthenticatedIdentities $true -IdentityPoolName  
CommonTests13
```

Output:

```
LoggedAt                : 8/12/2015 4:56:07 PM  
AllowUnauthenticatedIdentities : True  
DeveloperProviderName   :  
IdentityPoolId          : us-east-1:15d49393-ab16-431a-b26e-EXAMPLEGUID3  
IdentityPoolName        : CommonTests13  
OpenIdConnectProviderARNs : {}  
SupportedLoginProviders  : {}  
ResponseMetadata         : Amazon.Runtime.ResponseMetadata  
ContentLength            : 136  
HttpStatusCode           : OK
```

- Untuk detail API, lihat [CreateIdentityPool](#) di Referensi AWS Tools for PowerShell Cmdlet.

## Swift

### SDK untuk 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 [AWS Repositori Contoh Kode](#).

Buat kumpulan identitas baru.

```
/// Create a new identity pool and return its ID.
///
/// - Parameters:
///   - name: The name to give the new identity pool.
///
/// - Returns: A string containing the newly created pool's ID, or `nil`
///   if an error occurred.
///
func createIdentityPool(name: String) async throws -> String? {
    let cognitoInputCall = CreateIdentityPoolInput(developerProviderName:
"com.exampleco.CognitoIdentityDemo",
                                                    identityPoolName: name)

    let result = try await cognitoIdentityClient.createIdentityPool(input:
cognitoInputCall)
    guard let poolId = result.identityPoolId else {
        return nil
    }

    return poolId
}
```

- Untuk informasi selengkapnya, lihat [AWS panduan pengembang SDK untuk Swift](#).
- Untuk detail API, lihat referensi [CreateIdentityPool AWSSDK](#) untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **DeleteIdentityPool** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan `DeleteIdentityPool`.

## CLI

### AWS CLI

Untuk menghapus kumpulan identitas

`delete-identity-pool` Contoh berikut menghapus kumpulan identitas yang ditentukan.

Perintah:

```
aws cognito-identity delete-identity-pool \  
  --identity-pool-id "us-west-2:11111111-1111-1111-1111-111111111111"
```

Perintah ini tidak menghasilkan output.

- Untuk detail API, lihat [DeleteIdentityPool](#) di Referensi AWS CLI Perintah.

## Java

### SDK untuk Java 2.x

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;  
import software.amazon.awssdk.awscore.exception.AwsServiceException;  
import software.amazon.awssdk.regions.Region;  
import software.amazon.awssdk.services.cognitoidentity.CognitoIdentityClient;  
import  
  software.amazon.awssdk.services.cognitoidentity.model.DeleteIdentityPoolRequest;  
  
/**  
 * 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 DeleteIdentityPool {

    public static void main(String[] args) {
        final String usage = ""

            Usage:
                <identityPoolId>\s

            Where:
                identityPoolId - The Id value of your identity pool.
            """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String identityPoolId = args[0];
        CognitoIdentityClient cognitoIdClient = CognitoIdentityClient.builder()
            .region(Region.US_EAST_1)
            .credentialsProvider(ProfileCredentialsProvider.create())
            .build();

        deleteIdPool(cognitoIdClient, identityPoolId);
        cognitoIdClient.close();
    }

    public static void deleteIdPool(CognitoIdentityClient cognitoIdClient, String
identityPoolId) {
        try {

            DeleteIdentityPoolRequest identityPoolRequest =
DeleteIdentityPoolRequest.builder()
                .identityPoolId(identityPoolId)
                .build();

            cognitoIdClient.deleteIdentityPool(identityPoolRequest);
            System.out.println("Done");

        } catch (AwsServiceException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

```
}  
}
```

- Untuk detail API, lihat [DeleteIdentityPool](#) di Referensi AWS SDK for Java 2.x API.

## PowerShell

### Alat untuk PowerShell

Contoh 1: Menghapus Identity Pool tertentu.

```
Remove-CGIIIdentityPool -IdentityPoolId us-east-1:0de2af35-2988-4d0b-b22d-  
EXAMPLEGUID1
```

- Untuk detail API, lihat [DeleteIdentityPool](#) di Referensi AWS Tools for PowerShell Cmdlet.

## Swift

### SDK untuk 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 [AWS Repositori Contoh Kode](#).

Hapus kumpulan identitas yang ditentukan.

```
/// Delete the specified identity pool.  
///  
/// - Parameters:  
///   - id: The ID of the identity pool to delete.
```



```
///
func deleteIdentityPool(id: String) async throws {
    let input = DeleteIdentityPoolInput(
        identityPoolId: id
    )

    _ = try await cognitoIdentityClient.deleteIdentityPool(input: input)
}
```

- Untuk informasi selengkapnya, lihat [AWS panduan pengembang SDK untuk Swift](#).
- Untuk detail API, lihat referensi [DeleteIdentityPool AWSSDK](#) untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Gunakan **DescribeIdentityPool** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan `DescribeIdentityPool`.

### CLI

#### AWS CLI

Untuk menggambarkan kumpulan identitas

Contoh ini menjelaskan kumpulan identitas.

Perintah:

```
aws cognito-identity describe-identity-pool --identity-pool-id "us-west-2:11111111-1111-1111-1111-111111111111"
```

Output:

```
{
  "IdentityPoolId": "us-west-2:11111111-1111-1111-1111-111111111111",
  "IdentityPoolName": "MyIdentityPool",
  "AllowUnauthenticatedIdentities": false,
  "CognitoIdentityProviders": [
    {
```

```

        "ProviderName": "cognito-idp.us-west-2.amazonaws.com/us-
west-2_111111111",
        "ClientId": "3n4b5urk1ft4f13mg5e62d9ado",
        "ServerSideTokenCheck": false
    }
]
}

```

- Untuk detail API, lihat [DescribeIdentityPool](#) di Referensi AWS CLI Perintah.

## PowerShell

### Alat untuk PowerShell

Contoh 1: Mengambil informasi tentang Identity Pool tertentu dengan idnya.

```
Get-CGIIIdentityPool -IdentityPoolId us-east-1:0de2af35-2988-4d0b-b22d-
EXAMPLEGUID1
```

Output:

```

LoggedAt                : 8/12/2015 4:29:40 PM
AllowUnauthenticatedIdentities : True
DeveloperProviderName   :
IdentityPoolId          : us-east-1:0de2af35-2988-4d0b-b22d-EXAMPLEGUID1
IdentityPoolName        : CommonTests1
OpenIdConnectProviderARNs : {}
SupportedLoginProviders  : {}
ResponseMetadata        : Amazon.Runtime.ResponseMetadata
ContentLength           : 142
HttpStatusCode           : OK

```

- Untuk detail API, lihat [DescribeIdentityPool](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Gunakan **GetCredentialsForIdentity** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan `GetCredentialsForIdentity`.

## Java

### SDK untuk Java 2.x

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.cognitoidentity.CognitoIdentityClient;
import
    software.amazon.awssdk.services.cognitoidentity.model.GetCredentialsForIdentityRequest;
import
    software.amazon.awssdk.services.cognitoidentity.model.GetCredentialsForIdentityResponse;
import
    software.amazon.awssdk.services.cognitoidentityprovider.model.CognitoIdentityProviderException;

/**
 * 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 GetIdentityCredentials {
    public static void main(String[] args) {

        final String usage = ""

            Usage:
                <identityId>\s

            Where:
                identityId - The Id of an existing identity in the format
                REGION:GUID.
            """;

        if (args.length != 1) {
```

```
        System.out.println(usage);
        System.exit(1);
    }

    String identityId = args[0];
    CognitoIdentityClient cognitoClient = CognitoIdentityClient.builder()
        .region(Region.US_EAST_1)
        .build();

    getCredsForIdentity(cognitoClient, identityId);
    cognitoClient.close();
}

public static void getCredsForIdentity(CognitoIdentityClient cognitoClient,
String identityId) {
    try {
        GetCredentialsForIdentityRequest getCredentialsForIdentityRequest =
GetCredentialsForIdentityRequest
            .builder()
            .identityId(identityId)
            .build();

        GetCredentialsForIdentityResponse response = cognitoClient
            .getCredentialsForIdentity(getCredentialsForIdentityRequest);
        System.out.println(
            "Identity ID " + response.identityId() + ", Access key ID " +
response.credentials().accessKeyId());

    } catch (CognitoIdentityProviderException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
}
```

- Untuk detail API, lihat [GetCredentialsForIdentity](#) di Referensi AWS SDK for Java 2.x API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Gunakan **GetIdentityPoolRoles** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan `GetIdentityPoolRoles`.

### CLI

#### AWS CLI

Untuk mendapatkan peran kumpulan identitas

Contoh ini mendapatkan peran kumpulan identitas.

Perintah:

```
aws cognito-identity get-identity-pool-roles --identity-pool-id "us-west-2:11111111-1111-1111-1111-111111111111"
```

Output:

```
{
  "IdentityPoolId": "us-west-2:11111111-1111-1111-1111-111111111111",
  "Roles": {
    "authenticated": "arn:aws:iam::111111111111:role/Cognito_MyIdentityPoolAuth_Role",
    "unauthenticated": "arn:aws:iam::111111111111:role/Cognito_MyIdentityPoolUnauth_Role"
  }
}
```

- Untuk detail API, lihat [GetIdentityPoolRoles](#) di Referensi AWS CLI Perintah.

### PowerShell

#### Alat untuk PowerShell

Contoh 1: Mendapatkan informasi tentang peran untuk Identity Pool tertentu.

```
Get-CGIIIdentityPoolRole -IdentityPoolId us-east-1:0de2af35-2988-4d0b-b22d-EXAMPLEGUID1
```

Output:

```
LoggedAt      : 8/12/2015 4:33:51 PM
IdentityPoolId : us-east-1:0de2af35-2988-4d0b-b22d-EXAMPLEGUID1
Roles         : {[unauthenticated, arn:aws:iam::123456789012:role/
CommonTests1Role]}
ResponseMetadata : Amazon.Runtime.ResponseMetadata
ContentLength   : 165
HttpStatusCode  : OK
```

- Untuk detail API, lihat [GetIdentityPoolRoles](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Gunakan **ListIdentityPools** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan `ListIdentityPools`.

### CLI

#### AWS CLI

Untuk membuat daftar kumpulan identitas

Contoh ini mencantumkan kumpulan identitas. Ada maksimal 20 identitas yang terdaftar.

Perintah:

```
aws cognito-identity list-identity-pools --max-results 20
```

Output:

```
{
  "IdentityPools": [
    {
      "IdentityPoolId": "us-west-2:11111111-1111-1111-1111-111111111111",
      "IdentityPoolName": "MyIdentityPool"
    },
    {
      "IdentityPoolId": "us-west-2:11111111-1111-1111-1111-111111111111",
      "IdentityPoolName": "AnotherIdentityPool"
    }
  ]
}
```

```
    },
    {
        "IdentityPoolId": "us-west-2:11111111-1111-1111-1111-111111111111",
        "IdentityPoolName": "IdentityPoolRegionA"
    }
]
}
```

- Untuk detail API, lihat [ListIdentityPools](#) di Referensi AWS CLI Perintah.

## Java

### SDK untuk Java 2.x

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.cognitoidentity.CognitoIdentityClient;
import
    software.amazon.awssdk.services.cognitoidentity.model.ListIdentityPoolsRequest;
import
    software.amazon.awssdk.services.cognitoidentity.model.ListIdentityPoolsResponse;
import
    software.amazon.awssdk.services.cognitoidentityprovider.model.CognitoIdentityProviderException;

/**
 * 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 ListIdentityPools {
    public static void main(String[] args) {
        CognitoIdentityClient cognitoClient = CognitoIdentityClient.builder()
            .region(Region.US_EAST_1)
```

```
        .build();

        listIdPools(cognitoClient);
        cognitoClient.close();
    }

    public static void listIdPools(CognitoIdentityClient cognitoClient) {
        try {
            ListIdentityPoolsRequest poolsRequest =
                ListIdentityPoolsRequest.builder()
                    .maxResults(15)
                    .build();

            ListIdentityPoolsResponse response =
                cognitoClient.listIdentityPools(poolsRequest);
            response.identityPools().forEach(pool -> {
                System.out.println("Pool ID: " + pool.identityPoolId());
                System.out.println("Pool name: " + pool.identityPoolName());
            });

        } catch (CognitoIdentityProviderException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Untuk detail API, lihat [ListIdentityPools](#) di Referensi AWS SDK for Java 2.x API.

## PowerShell

### Alat untuk PowerShell

Contoh 1: Mengambil daftar Identity Pools yang ada.

```
Get-CGIIIdentityPoolList
```

Output:

```
IdentityPoolId
IdentityPoolName
```



```

-----
-----
us-east-1:0de2af35-2988-4d0b-b22d-EXAMPLEGUID1           CommonTests1
us-east-1:118d242d-204e-4b88-b803-EXAMPLEGUID2           Tests2
us-east-1:15d49393-ab16-431a-b26e-EXAMPLEGUID3           CommonTests13

```

- Untuk detail API, lihat [ListIdentityPools](#) di Referensi AWS Tools for PowerShell Cmdlet.

## Swift

### SDK untuk 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 [AWS Repositori Contoh Kode](#).

Temukan ID kumpulan identitas yang diberi namanya.

```

/// Return the ID of the identity pool with the specified name.
///
/// - Parameters:
///   - name: The name of the identity pool whose ID should be returned.
///
/// - Returns: A string containing the ID of the specified identity pool
///   or `nil` on error or if not found.
///
func getIdentityPoolID(name: String) async throws -> String? {
    var token: String? = nil

    // Iterate over the identity pools until a match is found.

    repeat {
        /// `token` is a value returned by `ListIdentityPools()` if the

```

```
    /// returned list of identity pools is only a partial list. You
    /// use the `token` to tell Amazon Cognito that you want to
    /// continue where you left off previously. If you specify `nil`
    /// or you don't provide the token, Amazon Cognito will start at
    /// the beginning.

    let listPoolsInput = ListIdentityPoolsInput(maxResults: 25,
nextToken: token)

    /// Read pages of identity pools from Cognito until one is found
    /// whose name matches the one specified in the `name` parameter.
    /// Return the matching pool's ID. Each time we ask for the next
    /// page of identity pools, we pass in the token given by the
    /// previous page.

    let output = try await cognitoIdentityClient.listIdentityPools(input:
listPoolsInput)

    if let identityPools = output.identityPools {
        for pool in identityPools {
            if pool.identityPoolName == name {
                return pool.identityPoolId!
            }
        }
    }

    token = output.nextToken
} while token != nil

return nil
}
```

Dapatkan ID dari kumpulan identitas yang ada atau buat jika belum ada.

```
/// Return the ID of the identity pool with the specified name.
///
/// - Parameters:
///   - name: The name of the identity pool whose ID should be returned
///
/// - Returns: A string containing the ID of the specified identity pool.
///   Returns `nil` if there's an error or if the pool isn't found.
///
```

```
public func getOrCreateIdentityPoolID(name: String) async throws -> String? {
    // See if the pool already exists. If it doesn't, create it.

    guard let poolId = try await self.getIdentityPoolID(name: name) else {
        return try await self.createIdentityPool(name: name)
    }

    return poolId
}
```

- Untuk informasi selengkapnya, lihat [AWS panduan pengembang SDK untuk Swift](#).
- Untuk detail API, lihat referensi [ListIdentityPools AWSSDK](#) untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Gunakan **SetIdentityPoolRoles** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan `SetIdentityPoolRoles`.

### CLI

#### AWS CLI

Untuk mengatur peran kumpulan identitas

`set-identity-pool-roles` Contoh berikut menetapkan peran kumpulan identitas.

```
aws cognito-identity set-identity-pool-roles \
  --identity-pool-id "us-west-2:11111111-1111-1111-1111-111111111111" \
  --roles authenticated="arn:aws:iam::111111111111:role/
Cognito_MyIdentityPoolAuth_Role"
```

- Untuk detail API, lihat [SetIdentityPoolRoles](#) di Referensi AWS CLI Perintah.

## PowerShell

### Alat untuk PowerShell

Contoh 1: Mengkonfigurasi Identity Pool tertentu untuk memiliki peran IAM yang tidak diautentikasi.

```
Set-CGIIIdentityPoolRole -IdentityPoolId us-east-1:0de2af35-2988-4d0b-b22d-EXAMPLEGUID1 -Role @{ "unauthenticated" = "arn:aws:iam::123456789012:role/CommonTests1Role" }
```

- Untuk detail API, lihat [SetIdentityPoolRoles](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

### Gunakan **UpdateIdentityPool** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan `UpdateIdentityPool`.

#### CLI

##### AWS CLI

Untuk memperbarui kumpulan identitas

Contoh ini memperbarui kumpulan identitas. Ini menetapkan nama untuk `MyIdentityPool`. Ia menambahkan Cognito sebagai penyedia identitas. Ini melarang identitas yang tidak diautentikasi.

Perintah:

```
aws cognito-identity update-identity-pool --identity-pool-id "us-west-2:11111111-1111-1111-1111-111111111111" --identity-pool-name "MyIdentityPool" --no-allow-unauthenticated-identities --cognito-identity-providers ProviderName="cognito-idp.us-west-2.amazonaws.com/us-west-2_11111111",ClientId="3n4b5urk1ft4fl3mg5e62d9ado",ServerSideTokenCheck=false
```

Output:

```
{
  "IdentityPoolId": "us-west-2:11111111-1111-1111-1111-111111111111",
  "IdentityPoolName": "MyIdentityPool",
  "AllowUnauthenticatedIdentities": false,
  "CognitoIdentityProviders": [
    {
      "ProviderName": "cognito-idp.us-west-2.amazonaws.com/us-west-2_1111111111",
      "ClientId": "3n4b5urk1ft4f13mg5e62d9ado",
      "ServerSideTokenCheck": false
    }
  ]
}
```

- Untuk detail API, lihat [UpdateIdentityPool](#) di Referensi AWS CLI Perintah.

## PowerShell

### Alat untuk PowerShell

Contoh 1: Memperbarui beberapa properti Identity Pool, dalam hal ini nama Identity Pool.

```
Update-CGIIIdentityPool -IdentityPoolId us-east-1:0de2af35-2988-4d0b-b22d-EXAMPLEGUID1 -IdentityPoolName NewPoolName
```

### Output:

```
LoggedAt                : 8/12/2015 4:53:33 PM
AllowUnauthenticatedIdentities : False
DeveloperProviderName   :
IdentityPoolId          : us-east-1:0de2af35-2988-4d0b-b22d-EXAMPLEGUID1
IdentityPoolName        : NewPoolName
OpenIdConnectProviderARNs : {}
SupportedLoginProviders  : {}
ResponseMetadata        : Amazon.Runtime.ResponseMetadata
ContentLength            : 135
HttpStatusCode           : OK
```

- Untuk detail API, lihat [UpdateIdentityPool](#) di Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Contoh lintas layanan untuk Identitas Amazon Cognito menggunakan SDK AWS

Contoh aplikasi berikut menggunakan AWS SDK untuk menggabungkan Identitas Amazon Cognito dengan yang lain. Layanan AWS Setiap contoh menyertakan tautan ke GitHub, di mana Anda dapat menemukan petunjuk tentang cara mengatur dan menjalankan aplikasi.

Contoh

- [Membangun aplikasi Amazon Transcribe](#)
- [Membuat aplikasi penjelajah Amazon Textract](#)

### Membangun aplikasi Amazon Transcribe

Contoh kode berikut ini menunjukkan cara menggunakan Amazon Transcribe untuk menyalin dan menampilkan rekaman suara di peramban.

JavaScript

SDK untuk JavaScript (v3)

Buat aplikasi yang menggunakan Amazon Transcribe untuk menyalin dan menampilkan rekaman suara di peramban. Aplikasi ini menggunakan dua bucket Amazon Simple Storage Service (Amazon S3), satu untuk meng-host kode aplikasi, dan satu lagi untuk menyimpan transkripsi. Aplikasi ini menggunakan kolam pengguna Amazon Cognito untuk mengautentikasi pengguna Anda. Pengguna yang diautentikasi memiliki izin AWS Identity and Access Management (IAM) untuk mengakses layanan yang diperlukan. AWS

Untuk kode sumber lengkap dan instruksi tentang cara mengatur dan menjalankan, lihat contoh lengkapnya di [GitHub](#).

Contoh ini juga tersedia di [panduan developer v3 AWS SDK for JavaScript](#).

Layanan yang digunakan dalam contoh ini

- Identitas Amazon Cognito

- Amazon S3
- Amazon Transcribe

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Membuat aplikasi penjelajah Amazon Textract

Contoh kode berikut ini menunjukkan cara menjelajahi output Amazon Textract melalui aplikasi interaktif.

### JavaScript

#### SDK untuk JavaScript (v3)

Menunjukkan cara menggunakan aplikasi AWS SDK for JavaScript untuk membangun aplikasi React yang menggunakan Amazon Textract untuk mengekstrak data dari gambar dokumen dan menampilkannya di halaman web interaktif. Contoh ini berjalan di peramban web dan memerlukan identitas Amazon Cognito yang diautentikasi sebagai kredensialnya. Contoh ini menggunakan Amazon Simple Storage Service (Amazon S3) untuk penyimpanan, dan untuk notifikasi, contoh ini mengambil polling antrian Amazon Simple Queue Service (Amazon SQS) yang berlangganan topik Amazon Simple Notification Service (Amazon SNS).

Untuk kode sumber lengkap dan instruksi tentang cara mengatur dan menjalankan, lihat contoh lengkapnya di [GitHub](#).

Layanan yang digunakan dalam contoh ini

- Identitas Amazon Cognito
- Amazon S3
- Amazon SNS
- Amazon SQS
- Amazon Textract

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

# Contoh kode untuk Penyedia Identitas Amazon Cognito menggunakan SDK AWS

Contoh kode berikut menunjukkan cara menggunakan Penyedia Identitas Amazon Cognito dengan kit pengembangan AWS perangkat lunak (SDK).

Tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Meskipun tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks pada skenario terkait dan contoh lintas layanan.

Skenario adalah contoh kode yang menunjukkan cara menyelesaikan tugas tertentu dengan memanggil beberapa fungsi dalam layanan yang sama.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Memulai

Halo Amazon Cognito

Contoh kode berikut menunjukkan cara memulai menggunakan Amazon Cognito.

## C++

### SDK untuk C++

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

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 cognito-idp)
```



```
# Set this project's name.
project("hello_cognito")

# Set the C++ standard to use to build this target.
# At least C++ 11 is required for the AWS SDK for C++.
set(CMAKE_CXX_STANDARD 11)

# Use the MSVC variable to determine if this is a Windows build.
set(WINDOWS_BUILD ${MSVC})

if (WINDOWS_BUILD) # Set the location where CMake can find the installed
  libraries for the AWS SDK.
  string(REPLACE ";" "/aws-cpp-sdk-all;" SYSTEM_MODULE_PATH
    "${CMAKE_SYSTEM_PREFIX_PATH}/aws-cpp-sdk-all")
  list(APPEND CMAKE_PREFIX_PATH ${SYSTEM_MODULE_PATH})
endif ()

# Find the AWS SDK for C++ package.
find_package(AWSSDK REQUIRED COMPONENTS ${SERVICE_COMPONENTS})

if (WINDOWS_BUILD AND AWSSDK_INSTALL_AS_SHARED_LIBS)
  # Copy relevant AWS SDK for C++ libraries into the current binary directory
  for running and debugging.

  # set(BIN_SUB_DIR "/Debug") # If you are building from the command line, you
  may need to uncomment this
                                # and set the proper subdirectory to the
  executables' location.

  AWSSDK_CPY_DYN_LIBS(SERVICE_COMPONENTS ""
    ${CMAKE_CURRENT_BINARY_DIR}${BIN_SUB_DIR})
endif ()

add_executable(${PROJECT_NAME}
  hello_cognito.cpp)

target_link_libraries(${PROJECT_NAME}
  ${AWSSDK_LINK_LIBRARIES})
```

Kode untuk file sumber hello\_cognito.cpp.

```
#include <aws/core/Aws.h>
```

```
#include <aws/cognito-idp/CognitoIdentityProviderClient.h>
#include <aws/cognito-idp/model/ListUserPoolsRequest.h>
#include <iostream>

/*
 * A "Hello Cognito" starter application which initializes an Amazon Cognito
 * client and lists the Amazon Cognito
 * user pools.
 *
 * main function
 *
 * Usage: 'hello_cognito'
 *
 */

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;
    {
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

        Aws::CognitoIdentityProvider::CognitoIdentityProviderClient
cognitoClient(clientConfig);

        Aws::String nextToken; // Used for pagination.
        std::vector<Aws::String> userPools;

        do {
            Aws::CognitoIdentityProvider::Model::ListUserPoolsRequest
listUserPoolsRequest;
            if (!nextToken.empty()) {
                listUserPoolsRequest.SetNextToken(nextToken);
            }

            Aws::CognitoIdentityProvider::Model::ListUserPoolsOutcome
listUserPoolsOutcome =
                cognitoClient.ListUserPools(listUserPoolsRequest);

            if (listUserPoolsOutcome.IsSuccess()) {
```

```
        for (auto &userPool:
listUserPoolsOutcome.GetResult().GetUserPools()) {

            userPools.push_back(userPool.GetName());
        }

        nextToken = listUserPoolsOutcome.GetResult().GetNextToken();
    } else {
        std::cerr << "ListUserPools error: " <<
listUserPoolsOutcome.GetError().GetMessage() << std::endl;
        result = 1;
        break;
    }

} while (!nextToken.empty());
std::cout << userPools.size() << " user pools found." << std::endl;
for (auto &userPool: userPools) {
    std::cout << "    user pool: " << userPool << std::endl;
}
}

Aws::ShutdownAPI(options); // Should only be called once.
return result;
}
```

- Untuk detail API, lihat [ListUserPools](#) di Referensi AWS SDK for C++ API.

## Go

### SDK untuk Go V2

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
package main
```

```
import (
    "context"
    "fmt"
    "log"

    "github.com/aws/aws-sdk-go-v2/aws"
    "github.com/aws/aws-sdk-go-v2/config"
    "github.com/aws/aws-sdk-go-v2/service/cognitoidentityprovider"
    "github.com/aws/aws-sdk-go-v2/service/cognitoidentityprovider/types"
)

// main uses the AWS SDK for Go V2 to create an Amazon Simple Notification
// Service
// (Amazon SNS) client and list the topics 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
    }
    cognitoClient := cognitoidentityprovider.NewFromConfig(sdkConfig)
    fmt.Println("Let's list the user pools for your account.")
    var pools []types.UserPoolDescriptionType
    paginator := cognitoidentityprovider.NewListUserPoolsPaginator(
        cognitoClient, &cognitoidentityprovider.ListUserPoolsInput{MaxResults:
aws.Int32(10)})
    for paginator.HasMorePages() {
        output, err := paginator.NextPage(context.TODO())
        if err != nil {
            log.Printf("Couldn't get user pools. Here's why: %v\n", err)
        } else {
            pools = append(pools, output.UserPools...)
        }
    }
    if len(pools) == 0 {
        fmt.Println("You don't have any user pools!")
    } else {
        for _, pool := range pools {
            fmt.Printf("\t\t%v: %v\n", *pool.Name, *pool.Id)
        }
    }
}
```

```
}  
}
```

- Untuk detail API, lihat [ListUserPools](#) di Referensi AWS SDK for Go API.

## Java

### SDK untuk Java 2.x

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
import software.amazon.awssdk.regions.Region;  
import  
    software.amazon.awssdk.services.cognitoidentityprovider.CognitoIdentityProviderClient;  
import  
    software.amazon.awssdk.services.cognitoidentityprovider.model.CognitoIdentityProviderException;  
import  
    software.amazon.awssdk.services.cognitoidentityprovider.model.ListUserPoolsResponse;  
import  
    software.amazon.awssdk.services.cognitoidentityprovider.model.ListUserPoolsRequest;  
  
/**  
 * 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 ListUserPools {  
    public static void main(String[] args) {  
        CognitoIdentityProviderClient cognitoClient =  
CognitoIdentityProviderClient.builder()  
            .region(Region.US_EAST_1)  
            .build();
```

```
        listAllUserPools(cognitoClient);
        cognitoClient.close();
    }

    public static void listAllUserPools(CognitoIdentityProviderClient
cognitoClient) {
        try {
            ListUserPoolsRequest request = ListUserPoolsRequest.builder()
                .maxResults(10)
                .build();

            ListUserPoolsResponse response =
cognitoClient.listUserPools(request);
            response.userPools().forEach(userpool -> {
                System.out.println("User pool " + userpool.name() + ", User ID "
+ userpool.id());
            });

        } catch (CognitoIdentityProviderException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Untuk detail API, lihat [ListUserPools](#) 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 [AWS Repositori Contoh Kode](#).

```
import {
    paginateListUserPools,
    CognitoIdentityProviderClient,
```

```
} from "@aws-sdk/client-cognito-identity-provider";

const client = new CognitoIdentityProviderClient({});

export const helloCognito = async () => {
  const paginator = paginateListUserPools({ client }, {});

  const userPoolNames = [];

  for await (const page of paginator) {
    const names = page.UserPools.map((pool) => pool.Name);
    userPoolNames.push(...names);
  }

  console.log("User pool names: ");
  console.log(userPoolNames.join("\n"));
  return userPoolNames;
};
```

- Untuk detail API, lihat [ListUserPools](#) di Referensi AWS SDK for JavaScript API.

### Contoh kode

- [Tindakan untuk Penyedia Identitas Amazon Cognito menggunakan SDK AWS](#)
  - [Gunakan AdminCreateUser dengan AWS SDK atau CLI](#)
  - [Gunakan AdminGetUser dengan AWS SDK atau CLI](#)
  - [Gunakan AdminInitiateAuth dengan AWS SDK atau CLI](#)
  - [Gunakan AdminRespondToAuthChallenge dengan AWS SDK atau CLI](#)
  - [Gunakan AdminSetUserPassword dengan AWS SDK atau CLI](#)
  - [Gunakan AssociateSoftwareToken dengan AWS SDK atau CLI](#)
  - [Gunakan ConfirmDevice dengan AWS SDK atau CLI](#)
  - [Gunakan ConfirmForgotPassword dengan AWS SDK atau CLI](#)
  - [Gunakan ConfirmSignUp dengan AWS SDK atau CLI](#)
  - [Gunakan CreateUserPool dengan AWS SDK atau CLI](#)
  - [Gunakan CreateUserPoolClient dengan AWS SDK atau CLI](#)
  - [Gunakan DeleteUser dengan AWS SDK atau CLI](#)
  - [Gunakan ForgotPassword dengan AWS SDK atau CLI](#)

- [Gunakan InitiateAuth dengan AWS SDK atau CLI](#)
- [Gunakan ListUserPools dengan AWS SDK atau CLI](#)
- [Gunakan ListUsers dengan AWS SDK atau CLI](#)
- [Gunakan ResendConfirmationCode dengan AWS SDK atau CLI](#)
- [Gunakan RespondToAuthChallenge dengan AWS SDK atau CLI](#)
- [Gunakan SignUp dengan AWS SDK atau CLI](#)
- [Gunakan UpdateUserPool dengan AWS SDK atau CLI](#)
- [Gunakan VerifySoftwareToken dengan AWS SDK atau CLI](#)
- [Skenario untuk Penyedia Identitas Amazon Cognito menggunakan SDK AWS](#)
  - [Secara otomatis mengonfirmasi pengguna Amazon Cognito yang dikenal dengan fungsi Lambda menggunakan SDK AWS](#)
  - [Secara otomatis memigrasikan pengguna Amazon Cognito yang dikenal dengan fungsi Lambda menggunakan SDK AWS](#)
  - [Mendaftar pengguna dengan kumpulan pengguna Amazon Cognito yang memerlukan MFA menggunakan SDK AWS](#)
  - [Menulis data aktivitas kustom dengan fungsi Lambda setelah autentikasi pengguna Amazon Cognito menggunakan SDK AWS](#)

## Tindakan untuk Penyedia Identitas Amazon Cognito menggunakan SDK AWS

Contoh kode berikut menunjukkan cara melakukan tindakan Penyedia Identitas Amazon Cognito individual dengan AWS SDK. Kutipan ini memanggil Amazon Cognito Identity Provider 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 Penyedia Identitas Amazon Cognito](#).

### Contoh

- [Gunakan AdminCreateUser dengan AWS SDK atau CLI](#)
- [Gunakan AdminGetUser dengan AWS SDK atau CLI](#)
- [Gunakan AdminInitiateAuth dengan AWS SDK atau CLI](#)



- [Gunakan AdminRespondToAuthChallenge dengan AWS SDK atau CLI](#)
- [Gunakan AdminSetUserPassword dengan AWS SDK atau CLI](#)
- [Gunakan AssociateSoftwareToken dengan AWS SDK atau CLI](#)
- [Gunakan ConfirmDevice dengan AWS SDK atau CLI](#)
- [Gunakan ConfirmForgotPassword dengan AWS SDK atau CLI](#)
- [Gunakan ConfirmSignUp dengan AWS SDK atau CLI](#)
- [Gunakan CreateUserPool dengan AWS SDK atau CLI](#)
- [Gunakan CreateUserPoolClient dengan AWS SDK atau CLI](#)
- [Gunakan DeleteUser dengan AWS SDK atau CLI](#)
- [Gunakan ForgotPassword dengan AWS SDK atau CLI](#)
- [Gunakan InitiateAuth dengan AWS SDK atau CLI](#)
- [Gunakan ListUserPools dengan AWS SDK atau CLI](#)
- [Gunakan ListUsers dengan AWS SDK atau CLI](#)
- [Gunakan ResendConfirmationCode dengan AWS SDK atau CLI](#)
- [Gunakan RespondToAuthChallenge dengan AWS SDK atau CLI](#)
- [Gunakan SignUp dengan AWS SDK atau CLI](#)
- [Gunakan UpdateUserPool dengan AWS SDK atau CLI](#)
- [Gunakan VerifySoftwareToken dengan AWS SDK atau CLI](#)

## Gunakan **AdminCreateUser** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan `AdminCreateUser`.

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:

- [Menulis data aktivitas khusus dengan fungsi Lambda setelah autentikasi pengguna Amazon Cognito](#)

CLI

AWS CLI

Untuk membuat pengguna

`admin-create-user` Contoh berikut membuat pengguna dengan pengaturan alamat email dan nomor telepon yang ditentukan.

```
aws cognito-idp admin-create-user \  
  --user-pool-id us-west-2_aaaaaaaaa \  
  --username diego \  
  --user-attributes Name=email,Value=diego@example.com \  
  Name=phone_number,Value="+15555551212" \  
  --message-action SUPPRESS
```

Output:

```
{  
  "User": {  
    "Username": "diego",  
    "Attributes": [  
      {  
        "Name": "sub",  
        "Value": "7325c1de-b05b-4f84-b321-9adc6e61f4a2"  
      },  
      {  
        "Name": "phone_number",  
        "Value": "+15555551212"  
      },  
      {  
        "Name": "email",  
        "Value": "diego@example.com"  
      }  
    ],  
    "UserCreateDate": 1548099495.428,  
    "UserLastModifiedDate": 1548099495.428,  
    "Enabled": true,  
    "UserStatus": "FORCE_CHANGE_PASSWORD"  
  }  
}
```

- Untuk detail API, lihat [AdminCreateUser](#) di Referensi AWS CLI Perintah.

## Go

## SDK untuk Go V2

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
type CognitoActions struct {
    CognitoClient *cognitoidentityprovider.Client
}

// AdminCreateUser uses administrator credentials to add a user to a user pool.
// This method leaves the user
// in a state that requires they enter a new password next time they sign in.
func (actor CognitoActions) AdminCreateUser(userPoolId string, userName string,
    userEmail string) error {
    _, err := actor.CognitoClient.AdminCreateUser(context.TODO(),
    &cognitoidentityprovider.AdminCreateUserInput{
        UserPoolId:    aws.String(userPoolId),
        Username:      aws.String(userName),
        MessageAction: types.MessageActionTypeSuppress,
        UserAttributes: []types.AttributeType{{Name: aws.String("email"), Value:
        aws.String(userEmail)}}},
    })
    if err != nil {
        var userExists *types.UsernameExistsException
        if errors.As(err, &userExists) {
            log.Printf("User %v already exists in the user pool.", userName)
            err = nil
        } else {
            log.Printf("Couldn't create user %v. Here's why: %v\n", userName, err)
        }
    }
    return err
}
```

- Untuk detail API, lihat [AdminCreateUser](#) di Referensi AWS SDK for Go API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Gunakan **AdminGetUser** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan `AdminGetUser`.

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:

- [Mendaftar pengguna dengan kumpulan pengguna yang membutuhkan MFA](#)

## .NET

### AWS SDK for .NET

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
/// <summary>
/// Get the specified user from an Amazon Cognito user pool with
administrator access.
/// </summary>
/// <param name="userName">The name of the user.</param>
/// <param name="poolId">The Id of the Amazon Cognito user pool.</param>
/// <returns>Async task.</returns>
public async Task<UserStatusType> GetAdminUserAsync(string userName, string
poolId)
{
    AdminGetUserRequest userRequest = new AdminGetUserRequest
    {
        Username = userName,
```

```

        UserPoolId = poolId,
    };

    var response = await _cognitoService.AdminGetUserAsync(userRequest);

    Console.WriteLine($"User status {response.UserStatus}");
    return response.UserStatus;
}

```

- Untuk detail API, lihat [AdminGetUser](#) di Referensi AWS SDK for .NET API.

## C++

### SDK untuk C++

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```

    Aws::Client::ClientConfiguration clientConfig;
    // Optional: Set to the AWS Region (overrides config file).
    // clientConfig.region = "us-east-1";

    Aws::CognitoIdentityProvider::CognitoIdentityProviderClient
client(clientConfig);

    Aws::CognitoIdentityProvider::Model::AdminGetUserRequest request;
    request.SetUsername(userName);
    request.SetUserPoolId(userPoolID);

    Aws::CognitoIdentityProvider::Model::AdminGetUserOutcome outcome =
        client.AdminGetUser(request);

    if (outcome.IsSuccess()) {
        std::cout << "The status for " << userName << " is " <<

    Aws::CognitoIdentityProvider::Model::UserStatusTypeMapper::GetNameForUserStatusType(
        outcome.GetResult().GetUserStatus()) << std::endl;

```

```
        std::cout << "Enabled is " << outcome.GetResult().GetEnabled() <<
std::endl;
    }
    else {
        std::cerr << "Error with CognitoIdentityProvider::AdminGetUser. "
        << outcome.GetError().GetMessage()
        << std::endl;
    }
}
```

- Untuk detail API, lihat [AdminGetUser](#) di Referensi AWS SDK for C++ API.

## CLI

### AWS CLI

Untuk mendapatkan pengguna

Contoh ini mendapatkan informasi tentang nama pengguna jane@example.com.

Perintah:

```
aws cognito-idp admin-get-user --user-pool-id us-west-2_aaaaaaaaa --username
jane@example.com
```

Output:

```
{
  "Username": "4320de44-2322-4620-999b-5e2e1c8df013",
  "Enabled": true,
  "UserStatus": "FORCE_CHANGE_PASSWORD",
  "UserCreateDate": 1548108509.537,
  "UserAttributes": [
    {
      "Name": "sub",
      "Value": "4320de44-2322-4620-999b-5e2e1c8df013"
    },
    {
      "Name": "email_verified",
      "Value": "true"
    },
    {
```

```
        "Name": "phone_number_verified",
        "Value": "true"
    },
    {
        "Name": "phone_number",
        "Value": "+01115551212"
    },
    {
        "Name": "email",
        "Value": "jane@example.com"
    }
],
"UserLastModifiedDate": 1548108509.537
}
```

- Untuk detail API, lihat [AdminGetUser](#) di Referensi AWS CLI Perintah.

## Java

### SDK untuk Java 2.x

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
public static void getAdminUser(CognitoIdentityProviderClient
identityProviderClient, String userName,
    String poolId) {
    try {
        AdminGetUserRequest userRequest = AdminGetUserRequest.builder()
            .username(userName)
            .userPoolId(poolId)
            .build();

        AdminGetUserResponse response =
identityProviderClient.adminGetUser(userRequest);
        System.out.println("User status " + response.userStatusAsString());

    } catch (CognitoIdentityProviderException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
    }
}
```

```
        System.exit(1);
    }
}
```

- Untuk detail API, lihat [AdminGetUser](#) 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 [AWS Repositori Contoh Kode](#).

```
const adminGetUser = ({ userPoolId, username }) => {
  const client = new CognitoIdentityProviderClient({});

  const command = new AdminGetUserCommand({
    UserPoolId: userPoolId,
    Username: username,
  });

  return client.send(command);
};
```

- Untuk detail API, lihat [AdminGetUser](#) di Referensi AWS SDK for JavaScript API.

## Kotlin

### SDK untuk Kotlin

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).



```
suspend fun getAdminUser(userNameVal: String?, poolIdVal: String?) {
    val userRequest = AdminGetUserRequest {
        username = userNameVal
        userPoolId = poolIdVal
    }

    CognitoIdentityProviderClient { region = "us-east-1" }.use
{ identityProviderClient ->
    val response = identityProviderClient.adminGetUser(userRequest)
    println("User status ${response.userStatus}")
}
}
```

- Untuk detail API, lihat [AdminGetUser](#) di AWS SDK untuk referensi API Kotlin.

## Python

### SDK untuk Python (Boto3)

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
class CognitoIdentityProviderWrapper:
    """Encapsulates Amazon Cognito actions"""

    def __init__(self, cognito_idp_client, user_pool_id, client_id,
client_secret=None):
        """
        :param cognito_idp_client: A Boto3 Amazon Cognito Identity Provider
client.
        :param user_pool_id: The ID of an existing Amazon Cognito user pool.
        :param client_id: The ID of a client application registered with the user
pool.
        :param client_secret: The client secret, if the client has a secret.
        """
        self.cognito_idp_client = cognito_idp_client
        self.user_pool_id = user_pool_id
```

```
self.client_id = client_id
self.client_secret = client_secret

def sign_up_user(self, user_name, password, user_email):
    """
    Signs up a new user with Amazon Cognito. This action prompts Amazon
    Cognito
    to send an email to the specified email address. The email contains a
    code that
    can be used to confirm the user.

    When the user already exists, the user status is checked to determine
    whether
    the user has been confirmed.

    :param user_name: The user name that identifies the new user.
    :param password: The password for the new user.
    :param user_email: The email address for the new user.
    :return: True when the user is already confirmed with Amazon Cognito.
             Otherwise, false.
    """
    try:
        kwargs = {
            "ClientId": self.client_id,
            "Username": user_name,
            "Password": password,
            "UserAttributes": [{"Name": "email", "Value": user_email}],
        }
        if self.client_secret is not None:
            kwargs["SecretHash"] = self._secret_hash(user_name)
        response = self.cognito_idp_client.sign_up(**kwargs)
        confirmed = response["UserConfirmed"]
    except ClientError as err:
        if err.response["Error"]["Code"] == "UsernameExistsException":
            response = self.cognito_idp_client.admin_get_user(
                UserPoolId=self.user_pool_id, Username=user_name
            )
            logger.warning(
                "User %s exists and is %s.", user_name,
                response["UserStatus"]
            )
            confirmed = response["UserStatus"] == "CONFIRMED"
        else:
```

```
        logger.error(  
            "Couldn't sign up %s. Here's why: %s: %s",  
            user_name,  
            err.response["Error"]["Code"],  
            err.response["Error"]["Message"],  
        )  
        raise  
    return confirmed
```

- Untuk detail API, lihat [AdminGetUser](#) di AWS SDK for Python (Boto3) Referensi API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Gunakan **AdminInitiateAuth** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan `AdminInitiateAuth`.

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:

- [Mendaftar pengguna dengan kumpulan pengguna yang membutuhkan MFA](#)

## .NET

### AWS SDK for .NET

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
/// <summary>  
/// Initiate an admin auth request.  
/// </summary>  
/// <param name="clientId">The client ID to use.</param>
```

```

/// <param name="userPoolId">The ID of the user pool.</param>
/// <param name="userName">The username to authenticate.</param>
/// <param name="password">The user's password.</param>
/// <returns>The session to use in challenge-response.</returns>
public async Task<string> AdminInitiateAuthAsync(string clientId, string
userPoolId, string userName, string password)
{
    var authParameters = new Dictionary<string, string>();
    authParameters.Add("USERNAME", userName);
    authParameters.Add("PASSWORD", password);

    var request = new AdminInitiateAuthRequest
    {
        ClientId = clientId,
        UserPoolId = userPoolId,
        AuthParameters = authParameters,
        AuthFlow = AuthFlowType.ADMIN_USER_PASSWORD_AUTH,
    };

    var response = await _cognitoService.AdminInitiateAuthAsync(request);
    return response.Session;
}

```

- Untuk detail API, lihat [AdminInitiateAuth](#) di Referensi AWS SDK for .NET API.

## C++

### SDK untuk C++

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```

Aws::Client::ClientConfiguration clientConfig;
// Optional: Set to the AWS Region (overrides config file).
// clientConfig.region = "us-east-1";

Aws::CognitoIdentityProvider::CognitoIdentityProviderClient
client(clientConfig);

```

```
Aws::CognitoIdentityProvider::Model::AdminInitiateAuthRequest request;
request.SetClientId(clientID);
request.SetUserPoolId(userPoolID);
request.AddAuthParameters("USERNAME", userName);
request.AddAuthParameters("PASSWORD", password);
request.SetAuthFlow(

Aws::CognitoIdentityProvider::Model::AuthFlowType::ADMIN_USER_PASSWORD_AUTH);

Aws::CognitoIdentityProvider::Model::AdminInitiateAuthOutcome outcome =
    client.AdminInitiateAuth(request);

if (outcome.IsSuccess()) {
    std::cout << "Call to AdminInitiateAuth was successful." << std::endl;
    sessionResult = outcome.GetResult().GetSession();
}
else {
    std::cerr << "Error with CognitoIdentityProvider::AdminInitiateAuth. "
        << outcome.GetError().GetMessage()
        << std::endl;
}
}
```

- Untuk detail API, lihat [AdminInitiateAuth](#) di Referensi AWS SDK for C++ API.

## CLI

### AWS CLI

Untuk memulai otorisasi

Contoh ini memulai otorisasi menggunakan alur ADMIN\_NO\_SRP\_AUTH untuk nama pengguna jane@example.com

Klien harus mengaktifkan API login untuk otentikasi berbasis server (ADMIN\_NO\_SRP\_AUTH).

Gunakan informasi sesi dalam nilai kembali untuk memanggil admin-respond-to-auth - challenge.

Perintah:

```
aws cognito-idp admin-initiate-auth --user-pool-id us-west-2_aaaaaaaaa --client-id 3n4b5urk1ft4f13mg5e62d9ado --auth-flow ADMIN_NO_SRP_AUTH --auth-parameters USERNAME=jane@example.com,PASSWORD=password
```

### Output:

```
{
  "ChallengeName": "NEW_PASSWORD_REQUIRED",
  "Session": "SESSION",
  "ChallengeParameters": {
    "USER_ID_FOR_SRP": "84514837-dcbc-4af1-abff-f3c109334894",
    "requiredAttributes": "[]",
    "userAttributes": "{\"email_verified\": \"true\", \"phone_number_verified\": \"true\", \"phone_number\": \"+01xxx5550100\", \"email\": \"jane@example.com\"}"
  }
}
```

- Untuk detail API, lihat [AdminInitiateAuth](#) di Referensi AWS CLI Perintah.

## Java

### SDK untuk Java 2.x

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
public static AdminInitiateAuthResponse
initiateAuth(CognitoIdentityProviderClient identityProviderClient,
             String clientId, String userName, String password, String userPoolId)
{
    try {
        Map<String, String> authParameters = new HashMap<>();
        authParameters.put("USERNAME", userName);
        authParameters.put("PASSWORD", password);

        AdminInitiateAuthRequest authRequest =
AdminInitiateAuthRequest.builder()
```

```
        .clientId(clientId)
        .userPoolId(userPoolId)
        .authParameters(authParameters)
        .authFlow(AuthFlowType.ADMIN_USER_PASSWORD_AUTH)
        .build();

        AdminInitiateAuthResponse response =
identityProviderClient.adminInitiateAuth(authRequest);
        System.out.println("Result Challenge is : " +
response.challengeName());
        return response;

    } catch (CognitoIdentityProviderException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }

    return null;
}
```

- Untuk detail API, lihat [AdminInitiateAuth](#) 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 [AWS Repositori Contoh Kode](#).

```
const adminInitiateAuth = ({ clientId, userPoolId, username, password }) => {
    const client = new CognitoIdentityProviderClient({});

    const command = new AdminInitiateAuthCommand({
        ClientId: clientId,
        UserPoolId: userPoolId,
        AuthFlow: AuthFlowType.ADMIN_USER_PASSWORD_AUTH,
        AuthParameters: { USERNAME: username, PASSWORD: password },
    });
```

```
return client.send(command);
};
```

- Untuk detail API, lihat [AdminInitiateAuth](#) di Referensi AWS SDK for JavaScript API.

## Kotlin

### SDK untuk Kotlin

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
suspend fun checkAuthMethod(clientIdVal: String, userNameVal: String,
passwordVal: String, userPoolIdVal: String): AdminInitiateAuthResponse {
    val authParas = mutableMapOf<String, String>()
    authParas["USERNAME"] = userNameVal
    authParas["PASSWORD"] = passwordVal

    val authRequest = AdminInitiateAuthRequest {
        clientId = clientIdVal
        userPoolId = userPoolIdVal
        authParameters = authParas
        authFlow = AuthFlowType.AdminUserPasswordAuth
    }

    CognitoIdentityProviderClient { region = "us-east-1" }.use
{ identityProviderClient ->
    val response = identityProviderClient.adminInitiateAuth(authRequest)
    println("Result Challenge is ${response.challengeName}")
    return response
}
}
```

- Untuk detail API, lihat [AdminInitiateAuth](#) di AWS SDK untuk referensi API Kotlin.



## Python

### SDK untuk Python (Boto3)

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
class CognitoIdentityProviderWrapper:
    """Encapsulates Amazon Cognito actions"""

    def __init__(self, cognito_idp_client, user_pool_id, client_id,
                 client_secret=None):
        """
        :param cognito_idp_client: A Boto3 Amazon Cognito Identity Provider
        client.
        :param user_pool_id: The ID of an existing Amazon Cognito user pool.
        :param client_id: The ID of a client application registered with the user
        pool.
        :param client_secret: The client secret, if the client has a secret.
        """
        self.cognito_idp_client = cognito_idp_client
        self.user_pool_id = user_pool_id
        self.client_id = client_id
        self.client_secret = client_secret

    def start_sign_in(self, user_name, password):
        """
        Starts the sign-in process for a user by using administrator credentials.
        This method of signing in is appropriate for code running on a secure
        server.

        If the user pool is configured to require MFA and this is the first sign-
        in
        for the user, Amazon Cognito returns a challenge response to set up an
        MFA application. When this occurs, this function gets an MFA secret from
        Amazon Cognito and returns it to the caller.

        :param user_name: The name of the user to sign in.
```

```

:param password: The user's password.
:return: The result of the sign-in attempt. When sign-in is successful,
this
        returns an access token that can be used to get AWS credentials.
Otherwise,
        Amazon Cognito returns a challenge to set up an MFA application,
        or a challenge to enter an MFA code from a registered MFA
application.
"""
    try:
        kwargs = {
            "UserPoolId": self.user_pool_id,
            "ClientId": self.client_id,
            "AuthFlow": "ADMIN_USER_PASSWORD_AUTH",
            "AuthParameters": {"USERNAME": user_name, "PASSWORD": password},
        }
        if self.client_secret is not None:
            kwargs["AuthParameters"]["SECRET_HASH"] =
self._secret_hash(user_name)
        response = self.cognito_idp_client.admin_initiate_auth(**kwargs)
        challenge_name = response.get("ChallengeName", None)
        if challenge_name == "MFA_SETUP":
            if (
                "SOFTWARE_TOKEN_MFA"
                in response["ChallengeParameters"]["MFAS_CAN_SETUP"]
            ):
                response.update(self.get_mfa_secret(response["Session"]))
            else:
                raise RuntimeError(
                    "The user pool requires MFA setup, but the user pool is
not "
                    "configured for TOTP MFA. This example requires TOTP
MFA."
                )
        except ClientError as err:
            logger.error(
                "Couldn't start sign in for %s. Here's why: %s: %s",
                user_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            response.pop("ResponseMetadata", None)

```

```
return response
```

- Untuk detail API, lihat [AdminInitiateAuth](#) di AWS SDK for Python (Boto3) Referensi API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Gunakan **AdminRespondToAuthChallenge** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan `AdminRespondToAuthChallenge`.

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:

- [Mendaftar pengguna dengan kumpulan pengguna yang membutuhkan MFA](#)

## .NET

### AWS SDK for .NET

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
/// <summary>
/// Respond to an admin authentication challenge.
/// </summary>
/// <param name="userName">The name of the user.</param>
/// <param name="clientId">The client ID.</param>
/// <param name="mfaCode">The multi-factor authentication code.</param>
/// <param name="session">The current application session.</param>
/// <param name="clientId">The user pool ID.</param>
/// <returns>The result of the authentication response.</returns>
public async Task<AuthenticationResultType> AdminRespondToAuthChallengeAsync(
    string userName,
```

```
        string clientId,
        string mfaCode,
        string session,
        string userPoolId)
    {
        Console.WriteLine("SOFTWARE_TOKEN_MFA challenge is generated");

        var challengeResponses = new Dictionary<string, string>();
        challengeResponses.Add("USERNAME", userName);
        challengeResponses.Add("SOFTWARE_TOKEN_MFA_CODE", mfaCode);

        var respondToAuthChallengeRequest = new
AdminRespondToAuthChallengeRequest
        {
            ChallengeName = ChallengeNameType.SOFTWARE_TOKEN_MFA,
            ClientId = clientId,
            ChallengeResponses = challengeResponses,
            Session = session,
            UserPoolId = userPoolId,
        };

        var response = await
_cognitoService.AdminRespondToAuthChallengeAsync(respondToAuthChallengeRequest);
        Console.WriteLine($"Response to Authentication
{response.AuthenticationResult.TokenType}");
        return response.AuthenticationResult;
    }
}
```

- Untuk detail API, lihat [AdminRespondToAuthChallenge](#) di Referensi AWS SDK for .NET API.

## C++

### SDK untuk C++

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```

    Aws::Client::ClientConfiguration clientConfig;
    // Optional: Set to the AWS Region (overrides config file).
    // clientConfig.region = "us-east-1";

    Aws::CognitoIdentityProvider::CognitoIdentityProviderClient
client(clientConfig);

    Aws::CognitoIdentityProvider::Model::AdminRespondToAuthChallengeRequest
request;
    request.AddChallengeResponses("USERNAME", userName);
    request.AddChallengeResponses("SOFTWARE_TOKEN_MFA_CODE", mfaCode);
    request.SetChallengeName(

Aws::CognitoIdentityProvider::Model::ChallengeNameType::SOFTWARE_TOKEN_MFA);
    request.SetClientId(clientID);
    request.SetUserPoolId(userPoolID);
    request.SetSession(session);

    Aws::CognitoIdentityProvider::Model::AdminRespondToAuthChallengeOutcome
outcome =
        client.AdminRespondToAuthChallenge(request);

    if (outcome.IsSuccess()) {
        std::cout << "Here is the response to the challenge.\n" <<

outcome.GetResult().GetAuthenticationResult().Jsonize().View().WriteReadable()
        << std::endl;

        accessToken =
outcome.GetResult().GetAuthenticationResult().GetAccessToken();
    }
    else {
        std::cerr << "Error with
CognitoIdentityProvider::AdminRespondToAuthChallenge. "
        << outcome.GetError().GetMessage()
        << std::endl;
        return false;
    }
}

```

- Untuk detail API, lihat [AdminRespondToAuthChallenge](#) di Referensi AWS SDK for C++ API.

## Java

### SDK untuk Java 2.x

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
// Respond to an authentication challenge.
public static void adminRespondToAuthChallenge(CognitoIdentityProviderClient
identityProviderClient,
    String userName, String clientId, String mfaCode, String session) {
    System.out.println("SOFTWARE_TOKEN_MFA challenge is generated");
    Map<String, String> challengeResponses = new HashMap<>();

    challengeResponses.put("USERNAME", userName);
    challengeResponses.put("SOFTWARE_TOKEN_MFA_CODE", mfaCode);

    AdminRespondToAuthChallengeRequest respondToAuthChallengeRequest =
AdminRespondToAuthChallengeRequest.builder()
        .challengeName(ChallengeNameType.SOFTWARE_TOKEN_MFA)
        .clientId(clientId)
        .challengeResponses(challengeResponses)
        .session(session)
        .build();

    AdminRespondToAuthChallengeResponse respondToAuthChallengeResult =
identityProviderClient
        .adminRespondToAuthChallenge(respondToAuthChallengeRequest);

    System.out.println("respondToAuthChallengeResult.getAuthenticationResult()"
        + respondToAuthChallengeResult.authenticationResult());
}
```

- Untuk detail API, lihat [AdminRespondToAuthChallenge](#) 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 [AWS Repositori Contoh Kode](#).

```
const adminRespondToAuthChallenge = ({
  userPoolId,
  clientId,
  username,
  totp,
  session,
}) => {
  const client = new CognitoIdentityProviderClient({});
  const command = new AdminRespondToAuthChallengeCommand({
    ChallengeName: ChallengeNameType.SOFTWARE_TOKEN_MFA,
    ChallengeResponses: {
      SOFTWARE_TOKEN_MFA_CODE: totp,
      USERNAME: username,
    },
    ClientId: clientId,
    UserPoolId: userPoolId,
    Session: session,
  });

  return client.send(command);
};
```

- Untuk detail API, lihat [AdminRespondToAuthChallenge](#) di Referensi AWS SDK for JavaScript API.

## Kotlin

### SDK untuk Kotlin

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
// Respond to an authentication challenge.
suspend fun adminRespondToAuthChallenge(userName: String, clientIdVal: String?,
    mfaCode: String, sessionVal: String?) {
    println("SOFTWARE_TOKEN_MFA challenge is generated")
    val challengeResponsesOb = mutableMapOf<String, String>()
    challengeResponsesOb["USERNAME"] = userName
    challengeResponsesOb["SOFTWARE_TOKEN_MFA_CODE"] = mfaCode

    val adminRespondToAuthChallengeRequest = AdminRespondToAuthChallengeRequest {
        challengeName = ChallengeNameType.SoftwareTokenMfa
        clientId = clientIdVal
        challengeResponses = challengeResponsesOb
        session = sessionVal
    }

    CognitoIdentityProviderClient { region = "us-east-1" }.use
    { identityProviderClient ->
        val respondToAuthChallengeResult =
            identityProviderClient.adminRespondToAuthChallenge(adminRespondToAuthChallengeRequest)
        println("respondToAuthChallengeResult.getAuthenticationResult()
        ${respondToAuthChallengeResult.authenticationResult}")
    }
}
```

- Untuk detail API, lihat [AdminRespondToAuthChallenge](#) di AWS SDK untuk referensi API Kotlin.



## Python

### SDK untuk Python (Boto3)

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

Menanggapi tantangan MFA dengan menyediakan kode yang dihasilkan oleh aplikasi MFA terkait.

```
class CognitoIdentityProviderWrapper:
    """Encapsulates Amazon Cognito actions"""

    def __init__(self, cognito_idp_client, user_pool_id, client_id,
                 client_secret=None):
        """
        :param cognito_idp_client: A Boto3 Amazon Cognito Identity Provider
        client.
        :param user_pool_id: The ID of an existing Amazon Cognito user pool.
        :param client_id: The ID of a client application registered with the user
        pool.
        :param client_secret: The client secret, if the client has a secret.
        """
        self.cognito_idp_client = cognito_idp_client
        self.user_pool_id = user_pool_id
        self.client_id = client_id
        self.client_secret = client_secret

    def respond_to_mfa_challenge(self, user_name, session, mfa_code):
        """
        Responds to a challenge for an MFA code. This completes the second step
        of
        a two-factor sign-in. When sign-in is successful, it returns an access
        token
        that can be used to get AWS credentials from Amazon Cognito.

        :param user_name: The name of the user who is signing in.
```

```
        :param session: Session information returned from a previous call to
initiate
                authentication.
        :param mfa_code: A code generated by the associated MFA application.
        :return: The result of the authentication. When successful, this contains
an
                access token for the user.
        """
        try:
            kwargs = {
                "UserPoolId": self.user_pool_id,
                "ClientId": self.client_id,
                "ChallengeName": "SOFTWARE_TOKEN_MFA",
                "Session": session,
                "ChallengeResponses": {
                    "USERNAME": user_name,
                    "SOFTWARE_TOKEN_MFA_CODE": mfa_code,
                },
            }
            if self.client_secret is not None:
                kwargs["ChallengeResponses"]["SECRET_HASH"] = self._secret_hash(
                    user_name
                )
            response =
self.cognito_idp_client.admin_respond_to_auth_challenge(**kwargs)
            auth_result = response["AuthenticationResult"]
        except ClientError as err:
            if err.response["Error"]["Code"] == "ExpiredCodeException":
                logger.warning(
                    "Your MFA code has expired or has been used already. You
might have "
                    "to wait a few seconds until your app shows you a new code."
                )
            else:
                logger.error(
                    "Couldn't respond to mfa challenge for %s. Here's why: %s:
%s",
                    user_name,
                    err.response["Error"]["Code"],
                    err.response["Error"]["Message"],
                )
                raise
        else:
            return auth_result
```

- Untuk detail API, lihat [AdminRespondToAuthChallenged](#) di AWS SDK for Python (Boto3) Referensi API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Gunakan **AdminSetUserPassword** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan `AdminSetUserPassword`.

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:

- [Menulis data aktivitas khusus dengan fungsi Lambda setelah autentikasi pengguna Amazon Cognito](#)

Go

SDK untuk Go V2

### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
type CognitoActions struct {
    CognitoClient *cognitoidentityprovider.Client
}
```

```
// AdminSetUserPassword uses administrator credentials to set a password for a
user without requiring a
```

```
// temporary password.
func (actor CognitoActions) AdminSetUserPassword(userPoolId string, userName
string, password string) error {
_, err := actor.CognitoClient.AdminSetUserPassword(context.TODO(),
&cognitoidentityprovider.AdminSetUserPasswordInput{
Password:  aws.String(password),
UserPoolId: aws.String(userPoolId),
Username:  aws.String(userName),
Permanent: true,
})
if err != nil {
var invalidPassword *types.InvalidPasswordException
if errors.As(err, &invalidPassword) {
log.Println(*invalidPassword.Message)
} else {
log.Printf("Couldn't set password for user %v. Here's why: %v\n", userName,
err)
}
}
return err
}
```

- Untuk detail API, lihat [AdminSetUserPassword](#) di Referensi AWS SDK for Go API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Gunakan **AssociateSoftwareToken** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan `AssociateSoftwareToken`.

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:

- [Mendaftar pengguna dengan kumpulan pengguna yang membutuhkan MFA](#)

## .NET

### AWS SDK for .NET

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
/// <summary>
/// Get an MFA token to authenticate the user with the authenticator.
/// </summary>
/// <param name="session">The session name.</param>
/// <returns>The session name.</returns>
public async Task<string> AssociateSoftwareTokenAsync(string session)
{
    var softwareTokenRequest = new AssociateSoftwareTokenRequest
    {
        Session = session,
    };

    var tokenResponse = await
        _cognitoService.AssociateSoftwareTokenAsync(softwareTokenRequest);
    var secretCode = tokenResponse.SecretCode;


    Console.WriteLine($"Use the following secret code to set up the
        authenticator: {secretCode}");

    return tokenResponse.Session;
}
```

- Untuk detail API, lihat [AssociateSoftwareToken](#) di Referensi AWS SDK for .NET API.

## C++

## SDK untuk C++

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
Aws::Client::ClientConfiguration clientConfig;
// Optional: Set to the AWS Region (overrides config file).
// clientConfig.region = "us-east-1";

Aws::CognitoIdentityProvider::CognitoIdentityProviderClient
client(clientConfig);

Aws::CognitoIdentityProvider::Model::AssociateSoftwareTokenRequest
request;
request.SetSession(session);

Aws::CognitoIdentityProvider::Model::AssociateSoftwareTokenOutcome
outcome =
    client.AssociateSoftwareToken(request);

if (outcome.IsSuccess()) {
    std::cout
        << "Enter this setup key into an authenticator app, for
example Google Authenticator."
        << std::endl;
    std::cout << "Setup key: " << outcome.GetResult().GetSecretCode()
        << std::endl;
#ifdef USING_QR
    printAsterisksLine();
    std::cout << "\nOr scan the QR code in the file '" << QR_CODE_PATH <<
        "." << std::endl;

    saveQRCode(std::string("otpauth://totp/" + userName + "?secret=" +
        outcome.GetResult().GetSecretCode());
#endif // USING_QR
    session = outcome.GetResult().GetSession();
```

```
    }
    else {
        std::cerr << "Error with
CognitoIdentityProvider::AssociateSoftwareToken. "
                << outcome.GetError().GetMessage()
                << std::endl;
        return false;
    }
}
```

- Untuk detail API, lihat [AssociateSoftwareToken](#) di Referensi AWS SDK for C++ API.

## Java

### SDK untuk Java 2.x

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
public static String getSecretForAppMFA(CognitoIdentityProviderClient
identityProviderClient, String session) {
    AssociateSoftwareTokenRequest softwareTokenRequest =
AssociateSoftwareTokenRequest.builder()
        .session(session)
        .build();

    AssociateSoftwareTokenResponse tokenResponse = identityProviderClient
        .associateSoftwareToken(softwareTokenRequest);
    String secretCode = tokenResponse.secretCode();
    System.out.println("Enter this token into Google Authenticator");
    System.out.println(secretCode);
    return tokenResponse.session();
}
```

- Untuk detail API, lihat [AssociateSoftwareToken](#) 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 [AWS Repositori Contoh Kode](#).

```
const associateSoftwareToken = (session) => {
  const client = new CognitoIdentityProviderClient({});
  const command = new AssociateSoftwareTokenCommand({
    Session: session,
  });

  return client.send(command);
};
```

- Untuk detail API, lihat [AssociateSoftwareToken](#) di Referensi AWS SDK for JavaScript API.

## Kotlin

### SDK untuk Kotlin

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
suspend fun getSecretForAppMFA(sessionVal: String?): String? {
  val softwareTokenRequest = AssociateSoftwareTokenRequest {
    session = sessionVal
  }

  CognitoIdentityProviderClient { region = "us-east-1" }.use
  { identityProviderClient ->
```



```

        val tokenResponse =
identityProviderClient.associateSoftwareToken(softwareTokenRequest)
        val secretCode = tokenResponse.secretCode
        println("Enter this token into Google Authenticator")
        println(secretCode)
        return tokenResponse.session
    }
}

```

- Untuk detail API, lihat [AssociateSoftwareToken](#) di AWS SDK untuk referensi API Kotlin.

## Python

### SDK untuk Python (Boto3)

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```

class CognitoIdentityProviderWrapper:
    """Encapsulates Amazon Cognito actions"""

    def __init__(self, cognito_idp_client, user_pool_id, client_id,
client_secret=None):
        """
        :param cognito_idp_client: A Boto3 Amazon Cognito Identity Provider
client.
        :param user_pool_id: The ID of an existing Amazon Cognito user pool.
        :param client_id: The ID of a client application registered with the user
pool.
        :param client_secret: The client secret, if the client has a secret.
        """
        self.cognito_idp_client = cognito_idp_client
        self.user_pool_id = user_pool_id
        self.client_id = client_id
        self.client_secret = client_secret

    def get_mfa_secret(self, session):

```

```
"""
    Gets a token that can be used to associate an MFA application with the
    user.

    :param session: Session information returned from a previous call to
    initiate
                    authentication.
    :return: An MFA token that can be used to set up an MFA application.
    """
    try:
        response =
self.cognito_idp_client.associate_software_token(Session=session)
    except ClientError as err:
        logger.error(
            "Couldn't get MFA secret. Here's why: %s: %s",
            err.response["Error"]["Code"],
            err.response["Error"]["Message"],
        )
        raise
    else:
        response.pop("ResponseMetadata", None)
        return response
```

- Untuk detail API, lihat [AssociateSoftwareToken](#) di AWS SDK for Python (Boto3) Referensi API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Gunakan **ConfirmDevice** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan `ConfirmDevice`.

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:

- [Mendaftar pengguna dengan kumpulan pengguna yang membutuhkan MFA](#)

## .NET

### AWS SDK for .NET

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
/// <summary>
/// Initiates and confirms tracking of the device.
/// </summary>
/// <param name="accessToken">The user's access token.</param>
/// <param name="deviceKey">The key of the device from Amazon Cognito.</
param>
/// <param name="deviceName">The device name.</param>
/// <returns></returns>
public async Task<bool> ConfirmDeviceAsync(string accessToken, string
deviceKey, string deviceName)
{
    var request = new ConfirmDeviceRequest
    {
        AccessToken = accessToken,
        DeviceKey = deviceKey,
        DeviceName = deviceName
    };

    var response = await _cognitoService.ConfirmDeviceAsync(request);
    return response.UserConfirmationNecessary;
}
```

- Untuk detail API, lihat [ConfirmDevice](#) di Referensi AWS SDK for .NET API.

## JavaScript

### SDK untuk JavaScript (v3)

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
const confirmDevice = ({ deviceKey, accessToken, passwordVerifier, salt }) => {
  const client = new CognitoIdentityProviderClient({});

  const command = new ConfirmDeviceCommand({
    DeviceKey: deviceKey,
    AccessToken: accessToken,
    DeviceSecretVerifierConfig: {
      PasswordVerifier: passwordVerifier,
      Salt: salt,
    },
  });

  return client.send(command);
};
```

- Untuk detail API, lihat [ConfirmDevice](#) di Referensi AWS SDK for JavaScript API.

## Python

### SDK untuk Python (Boto3)

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
class CognitoIdentityProviderWrapper:
    """Encapsulates Amazon Cognito actions"""
```

```

def __init__(self, cognito_idp_client, user_pool_id, client_id,
client_secret=None):
    """
    :param cognito_idp_client: A Boto3 Amazon Cognito Identity Provider
client.
    :param user_pool_id: The ID of an existing Amazon Cognito user pool.
    :param client_id: The ID of a client application registered with the user
pool.
    :param client_secret: The client secret, if the client has a secret.
    """
    self.cognito_idp_client = cognito_idp_client
    self.user_pool_id = user_pool_id
    self.client_id = client_id
    self.client_secret = client_secret

def confirm_mfa_device(
    self,
    user_name,
    device_key,
    device_group_key,
    device_password,
    access_token,
    aws_srp,
):
    """
    Confirms an MFA device to be tracked by Amazon Cognito. When a device is
tracked, its key and password can be used to sign in without requiring a
new
MFA code from the MFA application.

    :param user_name: The user that is associated with the device.
    :param device_key: The key of the device, returned by Amazon Cognito.
    :param device_group_key: The group key of the device, returned by Amazon
Cognito.
    :param device_password: The password that is associated with the device.
    :param access_token: The user's access token.
    :param aws_srp: A class that helps with Secure Remote Password (SRP)
calculations. The scenario associated with this example
uses
the warrant package.
    :return: True when the user must confirm the device. Otherwise, False.
When

```

```
        False, the device is automatically confirmed and tracked.
    """
    srp_helper = aws_srp.AWSSRP(
        username=user_name,
        password=device_password,
        pool_id="_",
        client_id=self.client_id,
        client_secret=None,
        client=self.cognito_idp_client,
    )
    device_and_pw = f"{device_group_key}{device_key}:{device_password}"
    device_and_pw_hash = aws_srp.hash_sha256(device_and_pw.encode("utf-8"))
    salt = aws_srp.pad_hex(aws_srp.get_random(16))
    x_value = aws_srp.hex_to_long(aws_srp.hex_hash(salt +
device_and_pw_hash))
    verifier = aws_srp.pad_hex(pow(srp_helper.val_g, x_value,
srp_helper.big_n))
    device_secret_verifier_config = {
        "PasswordVerifier": base64.standard_b64encode(
            bytearray.fromhex(verifier)
        ).decode("utf-8"),
        "Salt":
base64.standard_b64encode(bytearray.fromhex(salt)).decode("utf-8"),
    }
    try:
        response = self.cognito_idp_client.confirm_device(
            AccessToken=access_token,
            DeviceKey=device_key,
            DeviceSecretVerifierConfig=device_secret_verifier_config,
        )
        user_confirm = response["UserConfirmationNecessary"]
    except ClientError as err:
        logger.error(
            "Couldn't confirm mfa device %s. Here's why: %s: %s",
            device_key,
            err.response["Error"]["Code"],
            err.response["Error"]["Message"],
        )
        raise
    else:
        return user_confirm
```

- Untuk detail API, lihat [ConfirmDevice](#) di AWS SDK for Python (Boto3) Referensi API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Gunakan **ConfirmForgotPassword** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan `ConfirmForgotPassword`.

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:

- [Secara otomatis memigrasikan pengguna yang dikenal dengan fungsi Lambda](#)

## CLI

### AWS CLI

Untuk mengonfirmasi kata sandi yang terlupakan

Contoh ini mengonfirmasi kata sandi yang terlupakan untuk nama pengguna `diego@example.com`.

Perintah:

```
aws cognito-idp confirm-forgot-password --client-id 3n4b5urk1ft4f13mg5e62d9ado --username=diego@example.com --password PASSWORD --confirmation-code CONF_CODE
```

- Untuk detail API, lihat [ConfirmForgotPassword](#) di Referensi AWS CLI Perintah.

## Go

### SDK untuk Go V2

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
type CognitoActions struct {
    CognitoClient *cognitoidentityprovider.Client
}

// ConfirmForgotPassword confirms a user with a confirmation code and a new
password.
func (actor CognitoActions) ConfirmForgotPassword(clientId string, code string,
    userName string, password string) error {
    _, err := actor.CognitoClient.ConfirmForgotPassword(context.TODO(),
    &cognitoidentityprovider.ConfirmForgotPasswordInput{
        ClientId:      aws.String(clientId),
        ConfirmationCode: aws.String(code),
        Password:      aws.String(password),
        Username:      aws.String(userName),
    })
    if err != nil {
        var invalidPassword *types.InvalidPasswordException
        if errors.As(err, &invalidPassword) {
            log.Println(*invalidPassword.Message)
        } else {
            log.Printf("Couldn't confirm user %v. Here's why: %v", userName, err)
        }
    }
    return err
}
```

- Untuk detail API, lihat [ConfirmForgotPassword](#) di Referensi AWS SDK for Go API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Gunakan **ConfirmSignUp** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan `ConfirmSignUp`.




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:

- [Mendaftar pengguna dengan kumpulan pengguna yang membutuhkan MFA](#)

.NET

AWS SDK for .NET

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
/// <summary>
/// Confirm that the user has signed up.
/// </summary>
/// <param name="clientId">The Id of this application.</param>
/// <param name="code">The confirmation code sent to the user.</param>
/// <param name="userName">The username.</param>
/// <returns>True if successful.</returns>
public async Task<bool> ConfirmSignupAsync(string clientId, string code,
string userName)
{
    var signUpRequest = new ConfirmSignupRequest
    {
        ClientId = clientId,
        ConfirmationCode = code,
        Username = userName,
    };

    var response = await _cognitoService.ConfirmSignupAsync(signUpRequest);
    if (response.HttpStatusCode == HttpStatusCode.OK)
    {
        Console.WriteLine($"{userName} was confirmed");
        return true;
    }
    return false;
}
```

- Untuk detail API, lihat [ConfirmSignUp](#) di Referensi AWS SDK for .NET API.

## C++

### SDK untuk C++

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
Aws::Client::ClientConfiguration clientConfig;
// Optional: Set to the AWS Region (overrides config file).
// clientConfig.region = "us-east-1";

Aws::CognitoIdentityProvider::CognitoIdentityProviderClient
client(clientConfig);

Aws::CognitoIdentityProvider::Model::ConfirmSignUpRequest request;
request.SetClientId(clientID);
request.SetConfirmationCode(confirmationCode);
request.SetUsername(userName);

Aws::CognitoIdentityProvider::Model::ConfirmSignUpOutcome outcome =
    client.ConfirmSignUp(request);

if (outcome.IsSuccess()) {
    std::cout << "ConfirmSignup was Successful."
              << std::endl;
}
else {
    std::cerr << "Error with CognitoIdentityProvider::ConfirmSignUp. "
              << outcome.GetError().GetMessage()
              << std::endl;
    return false;
}
```

- Untuk detail API, lihat [ConfirmSignUp](#) di Referensi AWS SDK for C++ API.

## CLI

### AWS CLI

Untuk mengonfirmasi pendaftaran

Contoh ini mengonfirmasi pendaftaran untuk nama pengguna `diego@example.com`.

Perintah:

```
aws cognito-idp confirm-sign-up --client-id 3n4b5urk1ft4fl3mg5e62d9ado --
username=diego@example.com --confirmation-code CONF_CODE
```

- Untuk detail API, lihat [ConfirmSignUp](#) di Referensi AWS CLI Perintah.

## Java

### SDK untuk Java 2.x

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
public static void confirmSignUp(CognitoIdentityProviderClient
identityProviderClient, String clientId, String code,
    String userName) {
    try {
        ConfirmSignUpRequest signUpRequest = ConfirmSignUpRequest.builder()
            .clientId(clientId)
            .confirmationCode(code)
            .username(userName)
            .build();

        identityProviderClient.confirmSignUp(signUpRequest);
        System.out.println(userName + " was confirmed");
    }
}
```

```
    } catch (CognitoIdentityProviderException e) {  
        System.err.println(e.awsErrorDetails().errorMessage());  
        System.exit(1);  
    }  
}
```

- Untuk detail API, lihat [ConfirmSignUp](#) 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 [AWS Repositori Contoh Kode](#).

```
const confirmSignUp = ({ clientId, username, code }) => {  
    const client = new CognitoIdentityProviderClient({});  
  
    const command = new ConfirmSignUpCommand({  
        ClientId: clientId,  
        Username: username,  
        ConfirmationCode: code,  
    });  
  
    return client.send(command);  
};
```

- Untuk detail API, lihat [ConfirmSignUp](#) di Referensi AWS SDK for JavaScript API.

## Kotlin

### SDK untuk Kotlin

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
suspend fun confirmSignUp(clientIdVal: String?, codeVal: String?, userNameVal:
String?) {
    val signUpRequest = ConfirmSignUpRequest {
        clientId = clientIdVal
        confirmationCode = codeVal
        username = userNameVal
    }

    CognitoIdentityProviderClient { region = "us-east-1" }.use
{ identityProviderClient ->
    identityProviderClient.confirmSignUp(signUpRequest)
    println("$userNameVal was confirmed")
}
}
```

- Untuk detail API, lihat [ConfirmSignUp](#) di AWS SDK untuk referensi API Kotlin.

## Python

### SDK untuk Python (Boto3)

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
class CognitoIdentityProviderWrapper:
    """Encapsulates Amazon Cognito actions"""
```

```
def __init__(self, cognito_idp_client, user_pool_id, client_id,
client_secret=None):
    """
    :param cognito_idp_client: A Boto3 Amazon Cognito Identity Provider
client.
    :param user_pool_id: The ID of an existing Amazon Cognito user pool.
    :param client_id: The ID of a client application registered with the user
pool.
    :param client_secret: The client secret, if the client has a secret.
    """
    self.cognito_idp_client = cognito_idp_client
    self.user_pool_id = user_pool_id
    self.client_id = client_id
    self.client_secret = client_secret

def confirm_user_sign_up(self, user_name, confirmation_code):
    """
    Confirms a previously created user. A user must be confirmed before they
can sign in to Amazon Cognito.

    :param user_name: The name of the user to confirm.
    :param confirmation_code: The confirmation code sent to the user's
registered
                           email address.
    :return: True when the confirmation succeeds.
    """
    try:
        kwargs = {
            "ClientId": self.client_id,
            "Username": user_name,
            "ConfirmationCode": confirmation_code,
        }
        if self.client_secret is not None:
            kwargs["SecretHash"] = self._secret_hash(user_name)
        self.cognito_idp_client.confirm_sign_up(**kwargs)
    except ClientError as err:
        logger.error(
            "Couldn't confirm sign up for %s. Here's why: %s: %s",
            user_name,
            err.response["Error"]["Code"],
            err.response["Error"]["Message"],
        )
```

```
        raise
    else:
        return True
```

- Untuk detail API, lihat [ConfirmSignUp](#) di AWS SDK for Python (Boto3) Referensi API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Gunakan **CreateUserPool** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan `CreateUserPool`.

### CLI

#### AWS CLI

Untuk membuat kumpulan pengguna yang dikonfigurasi minimal

Contoh ini membuat kumpulan pengguna bernama `MyUserPool` menggunakan nilai default. Tidak ada atribut yang diperlukan dan tidak ada klien aplikasi. MFA dan keamanan tingkat lanjut dinonaktifkan.

Perintah:

```
aws cognito-idp create-user-pool --pool-name MyUserPool
```

Output:

```
{
  "UserPool": {
    "SchemaAttributes": [
      {
        "Name": "sub",
        "StringAttributeConstraints": {
          "MinLength": "1",
          "MaxLength": "2048"
        },
        "DeveloperOnlyAttribute": false,
```

```
    "Required": true,
    "AttributeDataType": "String",
    "Mutable": false
  },
  {
    "Name": "name",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "Name": "given_name",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "Name": "family_name",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "Name": "middle_name",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
```



```
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "Name": "nickname",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "Name": "preferred_username",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "Name": "profile",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "Name": "picture",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
```

```
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "Name": "website",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "Name": "email",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "AttributeDataType": "Boolean",
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "Name": "email_verified",
    "Mutable": true
  },
  {
    "Name": "gender",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
}
```

```
{
  "Name": "birthdate",
  "StringAttributeConstraints": {
    "MinLength": "10",
    "MaxLength": "10"
  },
  "DeveloperOnlyAttribute": false,
  "Required": false,
  "AttributeDataType": "String",
  "Mutable": true
},
{
  "Name": "zoneinfo",
  "StringAttributeConstraints": {
    "MinLength": "0",
    "MaxLength": "2048"
  },
  "DeveloperOnlyAttribute": false,
  "Required": false,
  "AttributeDataType": "String",
  "Mutable": true
},
{
  "Name": "locale",
  "StringAttributeConstraints": {
    "MinLength": "0",
    "MaxLength": "2048"
  },
  "DeveloperOnlyAttribute": false,
  "Required": false,
  "AttributeDataType": "String",
  "Mutable": true
},
{
  "Name": "phone_number",
  "StringAttributeConstraints": {
    "MinLength": "0",
    "MaxLength": "2048"
  },
  "DeveloperOnlyAttribute": false,
  "Required": false,
  "AttributeDataType": "String",
  "Mutable": true
},
}
```

```
{
  "AttributeDataType": "Boolean",
  "DeveloperOnlyAttribute": false,
  "Required": false,
  "Name": "phone_number_verified",
  "Mutable": true
},
{
  "Name": "address",
  "StringAttributeConstraints": {
    "MinLength": "0",
    "MaxLength": "2048"
  },
  "DeveloperOnlyAttribute": false,
  "Required": false,
  "AttributeDataType": "String",
  "Mutable": true
},
{
  "Name": "updated_at",
  "NumberAttributeConstraints": {
    "MinValue": "0"
  },
  "DeveloperOnlyAttribute": false,
  "Required": false,
  "AttributeDataType": "Number",
  "Mutable": true
}
],
"MfaConfiguration": "OFF",
"Name": "MyUserPool",
"LastModifiedDate": 1547833345.777,
"AdminCreateUserConfig": {
  "UnusedAccountValidityDays": 7,
  "AllowAdminCreateUserOnly": false
},
"EmailConfiguration": {},
"Policies": {
  "PasswordPolicy": {
    "RequireLowercase": true,
    "RequireSymbols": true,
    "RequireNumbers": true,
    "MinimumLength": 8,
    "RequireUppercase": true
  }
}
```

```
    }
  },
  "CreationDate": 1547833345.777,
  "EstimatedNumberOfUsers": 0,
  "Id": "us-west-2_aaaaaaaaa",
  "LambdaConfig": {}
}
}
```

Untuk membuat kumpulan pengguna dengan dua atribut yang diperlukan

Contoh ini membuat kumpulan pengguna MyUserPool. Pool dikonfigurasi untuk menerima email sebagai atribut nama pengguna. Ini juga menetapkan alamat sumber email ke alamat yang divalidasi menggunakan Amazon Simple Email Service.

Perintah:

```
aws cognito-idp create-user-pool --pool-name MyUserPool --username-attributes "email" --email-configuration=SourceArn="arn:aws:ses:us-east-1:111111111111:identity/jane@example.com",ReplyToEmailAddress="jane@example.com"
```

Output:

```
{
  "UserPool": {
    "SchemaAttributes": [
      {
        "Name": "sub",
        "StringAttributeConstraints": {
          "MinLength": "1",
          "MaxLength": "2048"
        },
        "DeveloperOnlyAttribute": false,
        "Required": true,
        "AttributeDataType": "String",
        "Mutable": false
      },
      {
        "Name": "name",
        "StringAttributeConstraints": {
          "MinLength": "0",
          "MaxLength": "2048"
        }
      }
    ]
  }
}
```

```
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "Name": "given_name",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "Name": "family_name",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "Name": "middle_name",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "Name": "nickname",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    }
  }
}
```

```
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "Name": "preferred_username",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "Name": "profile",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "Name": "picture",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "Name": "website",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    }
  }
}
```

```
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "Name": "email",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "AttributeDataType": "Boolean",
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "Name": "email_verified",
    "Mutable": true
  },
  {
    "Name": "gender",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "Name": "birthdate",
    "StringAttributeConstraints": {
      "MinLength": "10",
      "MaxLength": "10"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
```



```
    "Mutable": true
  },
  {
    "Name": "zoneinfo",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "Name": "locale",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "Name": "phone_number",
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "AttributeDataType": "Boolean",
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "Name": "phone_number_verified",
    "Mutable": true
  },
  {
    "Name": "address",
```

```
    "StringAttributeConstraints": {
      "MinLength": "0",
      "MaxLength": "2048"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "String",
    "Mutable": true
  },
  {
    "Name": "updated_at",
    "NumberAttributeConstraints": {
      "MinValue": "0"
    },
    "DeveloperOnlyAttribute": false,
    "Required": false,
    "AttributeDataType": "Number",
    "Mutable": true
  }
],
"MfaConfiguration": "OFF",
"Name": "MyUserPool",
"LastModifiedDate": 1547837788.189,
"AdminCreateUserConfig": {
  "UnusedAccountValidityDays": 7,
  "AllowAdminCreateUserOnly": false
},
"EmailConfiguration": {
  "ReplyToEmailAddress": "jane@example.com",
  "SourceArn": "arn:aws:ses:us-east-1:111111111111:identity/jane@example.com"
},
"Policies": {
  "PasswordPolicy": {
    "RequireLowercase": true,
    "RequireSymbols": true,
    "RequireNumbers": true,
    "MinimumLength": 8,
    "RequireUppercase": true
  }
},
"UsernameAttributes": [
  "email"
],
```

```
"CreationDate": 1547837788.189,  
"EstimatedNumberOfUsers": 0,  
"Id": "us-west-2_aaaaaaaa",  
"LambdaConfig": {}  
}  
}
```

- Untuk detail API, lihat [CreateUserPool](#) di Referensi AWS CLI Perintah.

## Java

### SDK untuk Java 2.x

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
import software.amazon.awssdk.regions.Region;  
import  
    software.amazon.awssdk.services.cognitoidentityprovider.CognitoIdentityProviderClient;  
import  
    software.amazon.awssdk.services.cognitoidentityprovider.model.CognitoIdentityProviderException;  
import  
    software.amazon.awssdk.services.cognitoidentityprovider.model.CreateUserPoolRequest;  
import  
    software.amazon.awssdk.services.cognitoidentityprovider.model.CreateUserPoolResponse;  
  
/**  
 * 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 CreateUserPool {  
    public static void main(String[] args) {  
  
        final String usage = ""
```

```
        Usage:
            <userPoolName>\s

        Where:
            userPoolName - The name to give your user pool when it's
created.

        """;

    if (args.length != 1) {
        System.out.println(usage);
        System.exit(1);
    }

    String userPoolName = args[0];
    CognitoIdentityProviderClient cognitoClient =
CognitoIdentityProviderClient.builder()
        .region(Region.US_EAST_1)
        .build();

    String id = createPool(cognitoClient, userPoolName);
    System.out.println("User pool ID: " + id);
    cognitoClient.close();
}

    public static String createPool(CognitoIdentityProviderClient cognitoClient,
String userPoolName) {
        try {
            CreateUserPoolRequest request = CreateUserPoolRequest.builder()
                .poolName(userPoolName)
                .build();

            CreateUserPoolResponse response =
cognitoClient.createUserPool(request);
            return response.userPool().id();

        } catch (CognitoIdentityProviderException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }
}
```

- Untuk detail API, lihat [CreateUserPool](#) di Referensi AWS SDK for Java 2.x API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Gunakan **CreateUserPoolClient** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan `CreateUserPoolClient`.

### CLI

#### AWS CLI

Untuk membuat klien kumpulan pengguna

Contoh ini membuat klien kumpulan pengguna baru dengan dua alur otorisasi eksplisit: `USER_PASSWORD_AUTH` dan `ADMIN_NO_SRP_AUTH`.

Perintah:

```
aws cognito-idp create-user-pool-client --user-pool-id us-west-2_aaaaaaaaaa
--client-name MyNewClient --no-generate-secret --explicit-auth-flows
"USER_PASSWORD_AUTH" "ADMIN_NO_SRP_AUTH"
```

Output:

```
{
  "UserPoolClient": {
    "UserPoolId": "us-west-2_aaaaaaaaaa",
    "ClientName": "MyNewClient",
    "ClientId": "6p3bs000no6a4ue1idruvd05ad",
    "LastModifiedDate": 1548697449.497,
    "CreationDate": 1548697449.497,
    "RefreshTokenValidity": 30,
    "ExplicitAuthFlows": [
      "USER_PASSWORD_AUTH",
      "ADMIN_NO_SRP_AUTH"
    ],
    "AllowedAuthFlowsUserPoolClient": false
  }
}
```

```
}  
}
```

- Untuk detail API, lihat [CreateUserPoolClient](#) di Referensi AWS CLI Perintah.

## Java

### SDK untuk Java 2.x

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
import software.amazon.awssdk.regions.Region;  
import  
    software.amazon.awssdk.services.cognitoidentityprovider.CognitoIdentityProviderClient;  
import  
    software.amazon.awssdk.services.cognitoidentityprovider.model.CognitoIdentityProviderException;  
import  
    software.amazon.awssdk.services.cognitoidentityprovider.model.CreateUserPoolClientRequest;  
import  
    software.amazon.awssdk.services.cognitoidentityprovider.model.CreateUserPoolClientResponse;  
  
/**  
 * A user pool client app is an application that authenticates with Amazon  
 * Cognito user pools.  
 * When you create a user pool, you can configure app clients that allow mobile  
 * or web applications  
 * to call API operations to authenticate users, manage user attributes and  
 * profiles,  
 * and implement sign-up and sign-in flows.  
 *  
 * 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 CreateUserPoolClient {
```

```
public static void main(String[] args) {
    final String usage = ""

        Usage:
            <clientName> <userPoolId>\s

        Where:
            clientName - The name for the user pool client to create.
            userPoolId - The ID for the user pool.
        """;

    if (args.length != 2) {
        System.out.println(usage);
        System.exit(1);
    }

    String clientName = args[0];
    String userPoolId = args[1];
    CognitoIdentityProviderClient cognitoClient =
CognitoIdentityProviderClient.builder()
        .region(Region.US_EAST_1)
        .build();

    createPoolClient(cognitoClient, clientName, userPoolId);
    cognitoClient.close();
}

public static void createPoolClient(CognitoIdentityProviderClient
cognitoClient, String clientName,
    String userPoolId) {
    try {
        CreateUserPoolClientRequest request =
CreateUserPoolClientRequest.builder()
            .clientName(clientName)
            .userPoolId(userPoolId)
            .build();

        CreateUserPoolClientResponse response =
cognitoClient.createUserPoolClient(request);
        System.out.println("User pool " +
response.userPoolClient().clientName() + " created. ID: "
            + response.userPoolClient().clientId());
    } catch (CognitoIdentityProviderException e) {
```

```
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
```

- Untuk detail API, lihat [CreateUserPoolClient](#) di Referensi AWS SDK for Java 2.x API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Gunakan **DeleteUser** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan `DeleteUser`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- [Secara otomatis mengonfirmasi pengguna yang dikenal dengan fungsi Lambda](#)
- [Secara otomatis memigrasikan pengguna yang dikenal dengan fungsi Lambda](#)
- [Menulis data aktivitas khusus dengan fungsi Lambda setelah autentikasi pengguna Amazon Cognito](#)

## C++

SDK untuk C++

### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
Aws::Client::ClientConfiguration clientConfig;
// Optional: Set to the AWS Region (overrides config file).
// clientConfig.region = "us-east-1";
```



```
Aws::CognitoIdentityProvider::CognitoIdentityProviderClient
client(clientConfig);

Aws::CognitoIdentityProvider::Model::DeleteUserRequest request;
request.SetAccessToken(accessToken);

Aws::CognitoIdentityProvider::Model::DeleteUserOutcome outcome =
    client.DeleteUser(request);

if (outcome.IsSuccess()) {
    std::cout << "The user " << userName << " was deleted."
              << std::endl;
}
else {
    std::cerr << "Error with CognitoIdentityProvider::DeleteUser. "
              << outcome.GetError().GetMessage()
              << std::endl;
}
```

- Untuk detail API, lihat [DeleteUser](#) di Referensi AWS SDK for C++ API.

## CLI

### AWS CLI

Untuk menghapus pengguna

Contoh ini menghapus pengguna.

Perintah:

```
aws cognito-idp delete-user --access-token ACCESS_TOKEN
```

- Untuk detail API, lihat [DeleteUser](#) di Referensi AWS CLI Perintah.

## Go

## SDK untuk Go V2

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
type CognitoActions struct {
    CognitoClient *cognitoidentityprovider.Client
}

// DeleteUser removes a user from the user pool.
func (actor CognitoActions) DeleteUser(userAccessToken string) error {
    _, err := actor.CognitoClient.DeleteUser(context.TODO(),
        &cognitoidentityprovider.DeleteUserInput{
            AccessToken: aws.String(userAccessToken),
        })
    if err != nil {
        log.Printf("Couldn't delete user. Here's why: %v\n", err)
    }
    return err
}
```

- Untuk detail API, lihat [DeleteUser](#) di Referensi AWS SDK for Go API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Gunakan **ForgotPassword** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan `ForgotPassword`.

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:

- [Secara otomatis memigrasikan pengguna yang dikenal dengan fungsi Lambda](#)

## CLI

### AWS CLI

Untuk memaksa perubahan kata sandi

`forgot-password` Contoh berikut mengirimkan pesan ke `jane@example.com` untuk mengubah kata sandi mereka.

```
aws cognito-idp forgot-password --client-id 38fjsnc484p94kpqsnet7mpld0 --username jane@example.com
```

Output:

```
{
  "CodeDeliveryDetails": {
    "Destination": "j***@e***.com",
    "DeliveryMedium": "EMAIL",
    "AttributeName": "email"
  }
}
```

- Untuk detail API, lihat [ForgotPassword](#) di Referensi AWS CLI Perintah.

## Go

### SDK untuk Go V2

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
type CognitoActions struct {
    CognitoClient *cognitoidentityprovider.Client
}

// ForgotPassword starts a password recovery flow for a user. This flow typically
// sends a confirmation code
// to the user's configured notification destination, such as email.
func (actor CognitoActions) ForgotPassword(clientId string, userName string)
(*types.CodeDeliveryDetailsType, error) {
    output, err := actor.CognitoClient.ForgotPassword(context.TODO(),
&cognitoidentityprovider.ForgotPasswordInput{
    ClientId: aws.String(clientId),
    Username: aws.String(userName),
})
    if err != nil {
        log.Printf("Couldn't start password reset for user '%v'. Here's why: %v\n",
userName, err)
    }
    return output.CodeDeliveryDetails, err
}
```

- Untuk detail API, lihat [ForgotPassword](#) di Referensi AWS SDK for Go API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Gunakan **InitiateAuth** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan `InitiateAuth`.

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:

- [Secara otomatis mengonfirmasi pengguna yang dikenal dengan fungsi Lambda](#)
- [Secara otomatis memigrasikan pengguna yang dikenal dengan fungsi Lambda](#)
- [Mendaftar pengguna dengan kumpulan pengguna yang membutuhkan MFA](#)

- [Menulis data aktivitas khusus dengan fungsi Lambda setelah autentikasi pengguna Amazon Cognito](#)

## .NET

### AWS SDK for .NET

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
/// <summary>
/// Initiate authorization.
/// </summary>
/// <param name="clientId">The client Id of the application.</param>
/// <param name="userName">The name of the user who is authenticating.</
param>
/// <param name="password">The password for the user who is authenticating.</
param>
/// <returns>The response from the initiate auth request.</returns>
public async Task<InitiateAuthResponse> InitiateAuthAsync(string clientId,
string userName, string password)
{
    var authParameters = new Dictionary<string, string>();
    authParameters.Add("USERNAME", userName);
    authParameters.Add("PASSWORD", password);

    var authRequest = new InitiateAuthRequest

    {
        ClientId = clientId,
        AuthParameters = authParameters,
        AuthFlow = AuthFlowType.USER_PASSWORD_AUTH,
    };

    var response = await _cognitoService.InitiateAuthAsync(authRequest);
    Console.WriteLine($"Result Challenge is : {response.ChallengeName}");


    return response;
}
```

```
}
```

- Untuk detail API, lihat [InitiateAuth](#) di Referensi AWS SDK for .NET API.

Go

SDK untuk Go V2

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
type CognitoActions struct {
    CognitoClient *cognitoidentityprovider.Client
}

// SignIn signs in a user to Amazon Cognito using a username and password
// authentication flow.
func (actor CognitoActions) SignIn(clientId string, userName string, password
string) (*types.AuthenticationResultType, error) {
    var authResult *types.AuthenticationResultType
    output, err := actor.CognitoClient.InitiateAuth(context.TODO(),
&cognitoidentityprovider.InitiateAuthInput{
    AuthFlow:      "USER_PASSWORD_AUTH",
    ClientId:      aws.String(clientId),
    AuthParameters: map[string]string{"USERNAME": userName, "PASSWORD": password},
    })
    if err != nil {
        var resetRequired *types.PasswordResetRequiredException
        if errors.As(err, &resetRequired) {
            log.Println(*resetRequired.Message)
        } else {
            log.Printf("Couldn't sign in user %v. Here's why: %v\n", userName, err)
        }
    } else {
        authResult = output.AuthenticationResult
    }
}
```

```
}  
  return authResult, err  
}
```

- Untuk detail API, lihat [InitiateAuth](#) 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 [AWS Repositori Contoh Kode](#).

```
const initiateAuth = ({ username, password, clientId }) => {  
  const client = new CognitoIdentityProviderClient({});  
  
  const command = new InitiateAuthCommand({  
    AuthFlow: AuthFlowType.USER_PASSWORD_AUTH,  
    AuthParameters: {  
      USERNAME: username,  
      PASSWORD: password,  
    },  
    ClientId: clientId,  
  });  
  
  return client.send(command);  
};
```

- Untuk detail API, lihat [InitiateAuth](#) di Referensi AWS SDK for JavaScript API.

## Python

### SDK untuk Python (Boto3)

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

Contoh ini menunjukkan cara memulai otentikasi dengan perangkat yang dilacak. Untuk menyelesaikan proses masuk, klien harus merespons tantangan Secure Remote Password (SRP) dengan benar.

```
class CognitoIdentityProviderWrapper:
    """Encapsulates Amazon Cognito actions"""

    def __init__(self, cognito_idp_client, user_pool_id, client_id,
client_secret=None):
        """
        :param cognito_idp_client: A Boto3 Amazon Cognito Identity Provider
client.
        :param user_pool_id: The ID of an existing Amazon Cognito user pool.
        :param client_id: The ID of a client application registered with the user
pool.
        :param client_secret: The client secret, if the client has a secret.
        """
        self.cognito_idp_client = cognito_idp_client
        self.user_pool_id = user_pool_id
        self.client_id = client_id
        self.client_secret = client_secret

    def sign_in_with_tracked_device(
        self,
        user_name,
        password,
        device_key,
        device_group_key,
        device_password,
        aws_srp,
    ):
```



```
"""
Signs in to Amazon Cognito as a user who has a tracked device. Signing in
with a tracked device lets a user sign in without entering a new MFA
code.
```

```
SRP
Signing in with a tracked device requires that the client respond to the
protocol. The scenario associated with this example uses the warrant
package
to help with SRP calculations.
```

```
For more information on SRP, see https://en.wikipedia.org/wiki/Secure\_Remote\_Password\_protocol.
```

```
:param user_name: The user that is associated with the device.
:param password: The user's password.
:param device_key: The key of a tracked device.
:param device_group_key: The group key of a tracked device.
:param device_password: The password that is associated with the device.
:param aws_srp: A class that helps with SRP calculations. The scenario
associated with this example uses the warrant package.
:return: The result of the authentication. When successful, this contains
an
access token for the user.
```

```
"""
try:
    srp_helper = aws_srp.AWSSRP(
        username=user_name,
        password=device_password,
        pool_id="_",
        client_id=self.client_id,
        client_secret=None,
        client=self.cognito_idp_client,
    )

    response_init = self.cognito_idp_client.initiate_auth(
        ClientId=self.client_id,
        AuthFlow="USER_PASSWORD_AUTH",
        AuthParameters={
            "USERNAME": user_name,
            "PASSWORD": password,
            "DEVICE_KEY": device_key,
        },
    )
```

```
        if response_init["ChallengeName"] != "DEVICE_SRP_AUTH":
            raise RuntimeError(
                f"Expected DEVICE_SRP_AUTH challenge but got
{response_init['ChallengeName']})."
            )

        auth_params = srp_helper.get_auth_params()
        auth_params["DEVICE_KEY"] = device_key
        response_auth = self.cognito_idp_client.respond_to_auth_challenge(
            ClientId=self.client_id,
            ChallengeName="DEVICE_SRP_AUTH",
            ChallengeResponses=auth_params,
        )
        if response_auth["ChallengeName"] != "DEVICE_PASSWORD_VERIFIER":
            raise RuntimeError(
                f"Expected DEVICE_PASSWORD_VERIFIER challenge but got "
                f"{response_init['ChallengeName']})."
            )

        challenge_params = response_auth["ChallengeParameters"]
        challenge_params["USER_ID_FOR_SRP"] = device_group_key + device_key
        cr = srp_helper.process_challenge(challenge_params, {"USERNAME":
user_name})
        cr["USERNAME"] = user_name
        cr["DEVICE_KEY"] = device_key
        response_verifier =
self.cognito_idp_client.respond_to_auth_challenge(
            ClientId=self.client_id,
            ChallengeName="DEVICE_PASSWORD_VERIFIER",
            ChallengeResponses=cr,
        )
        auth_tokens = response_verifier["AuthenticationResult"]
    except ClientError as err:
        logger.error(
            "Couldn't start client sign in for %s. Here's why: %s: %s",
            user_name,
            err.response["Error"]["Code"],
            err.response["Error"]["Message"],
        )
        raise
    else:
        return auth_tokens
```

- Untuk detail API, lihat [InitiateAuth](#) di AWS SDK for Python (Boto3) Referensi API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Gunakan **ListUserPools** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan `ListUserPools`.

.NET

AWS SDK for .NET

### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
/// <summary>
/// List the Amazon Cognito user pools for an account.
/// </summary>
/// <returns>A list of UserPoolDescriptionType objects.</returns>
public async Task<List<UserPoolDescriptionType>> ListUserPoolsAsync()
{
    var userPools = new List<UserPoolDescriptionType>();

    var userPoolsPaginator = _cognitoService.Paginators.ListUserPools(new
ListUserPoolsRequest());

    await foreach (var response in userPoolsPaginator.Responses)
    {
        userPools.AddRange(response.UserPools);
    }

    return userPools;
}
```

- Untuk detail API, lihat [ListUserPools](#) di Referensi AWS SDK for .NET API.

## CLI

### AWS CLI

Untuk daftar kumpulan pengguna

Contoh ini mencantumkan hingga 20 kumpulan pengguna.

Perintah:

```
aws cognito-idp list-user-pools --max-results 20
```

Output:

```
{
  "UserPools": [
    {
      "CreationDate": 1547763720.822,
      "LastModifiedDate": 1547763720.822,
      "LambdaConfig": {},
      "Id": "us-west-2_aaaaaaaaa",
      "Name": "MyUserPool"
    }
  ]
}
```

- Untuk detail API, lihat [ListUserPools](#) di Referensi AWS CLI Perintah.

## Go

### SDK untuk Go V2

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
package main

import (
    "context"
    "fmt"
    "log"

    "github.com/aws/aws-sdk-go-v2/aws"
    "github.com/aws/aws-sdk-go-v2/config"
    "github.com/aws/aws-sdk-go-v2/service/cognitoidentityprovider"
    "github.com/aws/aws-sdk-go-v2/service/cognitoidentityprovider/types"
)

// main uses the AWS SDK for Go V2 to create an Amazon Simple Notification
// Service
// (Amazon SNS) client and list the topics 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
    }
    cognitoClient := cognitoidentityprovider.NewFromConfig(sdkConfig)
    fmt.Println("Let's list the user pools for your account.")
    var pools []types.UserPoolDescriptionType
    paginator := cognitoidentityprovider.NewListUserPoolsPaginator(
        cognitoClient, &cognitoidentityprovider.ListUserPoolsInput{MaxResults:
aws.Int32(10)})
    for paginator.HasMorePages() {
        output, err := paginator.NextPage(context.TODO())
        if err != nil {
            log.Printf("Couldn't get user pools. Here's why: %v\n", err)
        } else {
            pools = append(pools, output.UserPools...)
        }
    }
    if len(pools) == 0 {
        fmt.Println("You don't have any user pools!")
    }
}
```

```
} else {
  for _, pool := range pools {
    fmt.Printf("\t\tv: %v\n", *pool.Name, *pool.Id)
  }
}
}
```

- Untuk detail API, lihat [ListUserPools](#) di Referensi AWS SDK for Go API.

## Java

### SDK untuk Java 2.x

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
import software.amazon.awssdk.regions.Region;
import
  software.amazon.awssdk.services.cognitoidentityprovider.CognitoIdentityProviderClient;
import
  software.amazon.awssdk.services.cognitoidentityprovider.model.CognitoIdentityProviderException;
import
  software.amazon.awssdk.services.cognitoidentityprovider.model.ListUserPoolsResponse;
import
  software.amazon.awssdk.services.cognitoidentityprovider.model.ListUserPoolsRequest;

/**
 * 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 ListUserPools {
    public static void main(String[] args) {
```

```
CognitoIdentityProviderClient cognitoClient =
CognitoIdentityProviderClient.builder()
    .region(Region.US_EAST_1)
    .build();

listAllUserPools(cognitoClient);
cognitoClient.close();
}

public static void listAllUserPools(CognitoIdentityProviderClient
cognitoClient) {
    try {
        ListUserPoolsRequest request = ListUserPoolsRequest.builder()
            .maxResults(10)
            .build();

        ListUserPoolsResponse response =
cognitoClient.listUserPools(request);
        response.userPools().forEach(userpool -> {
            System.out.println("User pool " + userpool.name() + ", User ID "
+ userpool.id());
        });

    } catch (CognitoIdentityProviderException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
}
```

- Untuk detail API, lihat [ListUserPools](#) di Referensi AWS SDK for Java 2.x API.

## Rust

### SDK untuk Rust

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```

async fn show_pools(client: &Client) -> Result<(), Error> {
    let response = client.list_user_pools().max_results(10).send().await?;
    let pools = response.user_pools();
    println!("User pools:");
    for pool in pools {
        println!(" ID:           {}", pool.id().unwrap_or_default());
        println!(" Name:           {}", pool.name().unwrap_or_default());
        println!(" Lambda Config:  {:?}", pool.lambda_config().unwrap());
        println!(
            " Last modified:  {}",
            pool.last_modified_date().unwrap().to_chrono_utc()?
        );
        println!(
            " Creation date:  {:?}",
            pool.creation_date().unwrap().to_chrono_utc()
        );
        println!();
    }
    println!("Next token: {}", response.next_token().unwrap_or_default());

    Ok(())
}

```

- Untuk detail API, lihat [ListUserPools](#) referensi AWS SDK for Rust API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Gunakan **ListUsers** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan `ListUsers`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- [Mendaftar pengguna dengan kumpulan pengguna yang membutuhkan MFA](#)



## .NET

### AWS SDK for .NET

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
/// <summary>
/// Get a list of users for the Amazon Cognito user pool.
/// </summary>
/// <param name="userPoolId">The user pool ID.</param>
/// <returns>A list of users.</returns>
public async Task<List<UserType>> ListUsersAsync(string userPoolId)
{
    var request = new ListUsersRequest
    {
        UserPoolId = userPoolId
    };

    var users = new List<UserType>();

    var usersPaginator = _cognitoService.Paginators.ListUsers(request);
    await foreach (var response in usersPaginator.Responses)
    {
        users.AddRange(response.Users);
    }

    return users;
}
```

- Untuk detail API, lihat [ListUsers](#) di Referensi AWS SDK for .NET API.

## CLI

### AWS CLI

Untuk daftar pengguna

Contoh ini mencantumkan hingga 20 pengguna.

Perintah:

```
aws cognito-idp list-users --user-pool-id us-west-2_aaaaaaaaa --limit 20
```

Output:

```
{
  "Users": [
    {
      "Username": "22704aa3-fc10-479a-97eb-2af5806bd327",
      "Enabled": true,
      "UserStatus": "FORCE_CHANGE_PASSWORD",
      "UserCreateDate": 1548089817.683,
      "UserLastModifiedDate": 1548089817.683,
      "Attributes": [
        {
          "Name": "sub",
          "Value": "22704aa3-fc10-479a-97eb-2af5806bd327"
        },
        {
          "Name": "email_verified",
          "Value": "true"
        },
        {
          "Name": "email",
          "Value": "mary@example.com"
        }
      ]
    }
  ]
}
```

- Untuk detail API, lihat [ListUsers](#) di Referensi AWS CLI Perintah.

## Java

### SDK untuk Java 2.x

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
import software.amazon.awssdk.regions.Region;
import
    software.amazon.awssdk.services.cognitoidentityprovider.CognitoIdentityProviderClient;
import
    software.amazon.awssdk.services.cognitoidentityprovider.model.CognitoIdentityProviderException;
import
    software.amazon.awssdk.services.cognitoidentityprovider.model.ListUsersRequest;
import
    software.amazon.awssdk.services.cognitoidentityprovider.model.ListUsersResponse;

/**
 * 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) {

        final String usage = ""

            Usage:
                <userPoolId>\s

            Where:
                userPoolId - The ID given to your user pool when it's
            created.

        """;
```

```
        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String userPoolId = args[0];
        CognitoIdentityProviderClient cognitoClient =
CognitoIdentityProviderClient.builder()
            .region(Region.US_EAST_1)
            .build();

        listAllUsers(cognitoClient, userPoolId);
        listUsersFilter(cognitoClient, userPoolId);
        cognitoClient.close();
    }

    public static void listAllUsers(CognitoIdentityProviderClient cognitoClient,
String userPoolId) {
        try {
            ListUsersRequest usersRequest = ListUsersRequest.builder()
                .userPoolId(userPoolId)
                .build();

            ListUsersResponse response = cognitoClient.listUsers(usersRequest);
            response.users().forEach(user -> {
                System.out.println("User " + user.username() + " Status " +
user.userStatus() + " Created "
                    + user.userCreateDate());
            });

        } catch (CognitoIdentityProviderException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    // Shows how to list users by using a filter.
    public static void listUsersFilter(CognitoIdentityProviderClient
cognitoClient, String userPoolId) {

        try {
            String filter = "email = \"tblue@noserver.com\"";
            ListUsersRequest usersRequest = ListUsersRequest.builder()
                .userPoolId(userPoolId)
```

```
        .filter(filter)
        .build();

        ListUsersResponse response = cognitoClient.listUsers(usersRequest);
        response.users().forEach(user -> {
            System.out.println("User with filter applied " + user.username()
+ " Status " + user.userStatus()
            + " Created " + user.userCreateDate());
        });

    } catch (CognitoIdentityProviderException 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 [AWS Repositori Contoh Kode](#).

```
const listUsers = ({ userPoolId }) => {
    const client = new CognitoIdentityProviderClient({});

    const command = new ListUsersCommand({
        UserPoolId: userPoolId,
    });

    return client.send(command);
};
```

- Untuk detail API, lihat [ListUsers](#) di Referensi AWS SDK for JavaScript API.

## Kotlin

### SDK untuk Kotlin

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
suspend fun listAllUsers(userPoolId: String) {  
  
    val request = ListUsersRequest {  
        this.userPoolId = userPoolId  
    }  
  
    CognitoIdentityProviderClient { region = "us-east-1" }.use { cognitoClient ->  
        val response = cognitoClient.listUsers(request)  
        response.users?.forEach { user ->  
            println("The user name is ${user.username}")  
        }  
    }  
}
```

- Untuk detail API, lihat [ListUsers](#) di AWS SDK untuk referensi API Kotlin.

## Python

### SDK untuk Python (Boto3)

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
class CognitoIdentityProviderWrapper:
    """Encapsulates Amazon Cognito actions"""

    def __init__(self, cognito_idp_client, user_pool_id, client_id,
client_secret=None):
        """
        :param cognito_idp_client: A Boto3 Amazon Cognito Identity Provider
client.
        :param user_pool_id: The ID of an existing Amazon Cognito user pool.
        :param client_id: The ID of a client application registered with the user
pool.
        :param client_secret: The client secret, if the client has a secret.
        """
        self.cognito_idp_client = cognito_idp_client
        self.user_pool_id = user_pool_id
        self.client_id = client_id
        self.client_secret = client_secret

    def list_users(self):
        """
        Returns a list of the users in the current user pool.

        :return: The list of users.
        """
        try:
            response =
self.cognito_idp_client.list_users(UserPoolId=self.user_pool_id)
            users = response["Users"]
        except ClientError as err:
            logger.error(
                "Couldn't list users for %s. Here's why: %s: %s",
                self.user_pool_id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return users
```

- Untuk detail API, lihat [ListUsers](#) di AWS SDK for Python (Boto3) Referensi API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Gunakan **ResendConfirmationCode** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan `ResendConfirmationCode`.

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:

- [Mendaftar pengguna dengan kumpulan pengguna yang membutuhkan MFA](#)

### .NET

#### AWS SDK for .NET

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
/// <summary>
/// Send a new confirmation code to a user.
/// </summary>
/// <param name="clientId">The Id of the client application.</param>
/// <param name="userName">The username of user who will receive the code.</
param>
/// <returns>The delivery details.</returns>
public async Task<CodeDeliveryDetailsType> ResendConfirmationCodeAsync(string
clientId, string userName)
{
    var codeRequest = new ResendConfirmationCodeRequest
    {
        ClientId = clientId,
        Username = userName,
    };

    var response = await
_cognitoService.ResendConfirmationCodeAsync(codeRequest);
```



```
        Console.WriteLine($"Method of delivery is  
{response.CodeDeliveryDetails.DeliveryMedium}");  
  
        return response.CodeDeliveryDetails;  
    }  
}
```

- Untuk detail API, lihat [ResendConfirmationCode](#) di Referensi AWS SDK for .NET API.

## C++

### SDK untuk C++

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
Aws::Client::ClientConfiguration clientConfig;  
// Optional: Set to the AWS Region (overrides config file).  
// clientConfig.region = "us-east-1";  
  
Aws::CognitoIdentityProvider::CognitoIdentityProviderClient  
client(clientConfig);  
  
Aws::CognitoIdentityProvider::Model::ResendConfirmationCodeRequest  
request;  
request.SetUsername(userName);  
request.SetClientId(clientID);  
  
Aws::CognitoIdentityProvider::Model::ResendConfirmationCodeOutcome  
outcome =  
    client.ResendConfirmationCode(request);  
  
if (outcome.IsSuccess()) {  
    std::cout  
        << "CognitoIdentityProvider::ResendConfirmationCode was  
successful."  
        << std::endl;  
}
```

```
    }
    else {
        std::cerr << "Error with
CognitoIdentityProvider::ResendConfirmationCode. "
                << outcome.GetError().GetMessage()
                << std::endl;
        return false;
    }
}
```

- Untuk detail API, lihat [ResendConfirmationCode](#) di Referensi AWS SDK for C++ API.

## CLI

### AWS CLI

Untuk mengirim ulang kode konfirmasi

`resend-confirmation-code` Contoh berikut mengirimkan kode konfirmasi kepada pengguna `jane`.

```
aws cognito-idp resend-confirmation-code \
  --client-id 12a3b456c7de890f11g123hijk \
  --username jane
```

Output:

```
{
  "CodeDeliveryDetails": {
    "Destination": "j***@e***.com",
    "DeliveryMedium": "EMAIL",
    "AttributeName": "email"
  }
}
```

Untuk informasi selengkapnya, lihat [Mendaftar dan mengonfirmasi akun pengguna](#) di Panduan Pengembang Amazon Cognito.

- Untuk detail API, lihat [ResendConfirmationCode](#) di Referensi AWS CLI Perintah.

## Java

### SDK untuk Java 2.x

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
public static void resendConfirmationCode(CognitoIdentityProviderClient
identityProviderClient, String clientId,
    String userName) {
    try {
        ResendConfirmationCodeRequest codeRequest =
ResendConfirmationCodeRequest.builder()
            .clientId(clientId)
            .username(userName)
            .build();

        ResendConfirmationCodeResponse response =
identityProviderClient.resendConfirmationCode(codeRequest);
        System.out.println("Method of delivery is " +
response.codeDeliveryDetails().deliveryMediumAsString());

    } catch (CognitoIdentityProviderException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
```

- Untuk detail API, lihat [ResendConfirmationCode](#) 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 [AWS Repositori Contoh Kode](#).

```
const resendConfirmationCode = ({ clientId, username }) => {
  const client = new CognitoIdentityProviderClient({});

  const command = new ResendConfirmationCodeCommand({
    ClientId: clientId,
    Username: username,
  });

  return client.send(command);
};
```

- Untuk detail API, lihat [ResendConfirmationCode](#) di Referensi AWS SDK for JavaScript API.

## Kotlin

### SDK untuk Kotlin

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
suspend fun resendConfirmationCode(clientIdVal: String?, userNameVal: String?) {
  val codeRequest = ResendConfirmationCodeRequest {
    clientId = clientIdVal
    username = userNameVal
  }
}
```

```

CognitoIdentityProviderClient { region = "us-east-1" }.use
{ identityProviderClient ->
    val response = identityProviderClient.resendConfirmationCode(codeRequest)
    println("Method of delivery is " +
(response.codeDeliveryDetails?.deliveryMedium))
    }
}

```

- Untuk detail API, lihat [ResendConfirmationCode](#) di AWS SDK untuk referensi API Kotlin.

## Python

### SDK untuk Python (Boto3)

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```

class CognitoIdentityProviderWrapper:
    """Encapsulates Amazon Cognito actions"""

    def __init__(self, cognito_idp_client, user_pool_id, client_id,
client_secret=None):
        """
        :param cognito_idp_client: A Boto3 Amazon Cognito Identity Provider
client.
        :param user_pool_id: The ID of an existing Amazon Cognito user pool.
        :param client_id: The ID of a client application registered with the user
pool.
        :param client_secret: The client secret, if the client has a secret.
        """
        self.cognito_idp_client = cognito_idp_client
        self.user_pool_id = user_pool_id
        self.client_id = client_id
        self.client_secret = client_secret

    def resend_confirmation(self, user_name):
        """

```

```
Prompts Amazon Cognito to resend an email with a new confirmation code.

:param user_name: The name of the user who will receive the email.
:return: Delivery information about where the email is sent.
"""
try:
    kwargs = {"ClientId": self.client_id, "Username": user_name}
    if self.client_secret is not None:
        kwargs["SecretHash"] = self._secret_hash(user_name)
    response = self.cognito_idp_client.resend_confirmation_code(**kwargs)
    delivery = response["CodeDeliveryDetails"]
except ClientError as err:
    logger.error(
        "Couldn't resend confirmation to %s. Here's why: %s: %s",
        user_name,
        err.response["Error"]["Code"],
        err.response["Error"]["Message"],
    )
    raise
else:
    return delivery
```

- Untuk detail API, lihat [ResendConfirmationCode](#) di AWS SDK for Python (Boto3) Referensi API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Gunakan **RespondToAuthChallenge** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan `RespondToAuthChallenge`.

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:

- [Mendaftar pengguna dengan kumpulan pengguna yang membutuhkan MFA](#)

## CLI

### AWS CLI

Untuk menanggapi tantangan otorisasi

Contoh ini menanggapi tantangan otorisasi yang dimulai dengan `initiate-auth`. Ini adalah respons terhadap tantangan `NEW_PASSWORD_REQUIRED`. Ini menetapkan kata sandi untuk pengguna `jane@example.com`.

Perintah:

```
aws cognito-idp respond-to-auth-challenge --client-id 3n4b5urk1ft4f13mg5e62d9ado
--challenge-name NEW_PASSWORD_REQUIRED --challenge-responses
USERNAME=jane@example.com,NEW_PASSWORD="password" --session "SESSION_TOKEN"
```

Output:

```
{
  "ChallengeParameters": {},
  "AuthenticationResult": {
    "AccessToken": "ACCESS_TOKEN",
    "ExpiresIn": 3600,
    "TokenType": "Bearer",
    "RefreshToken": "REFRESH_TOKEN",
    "IdToken": "ID_TOKEN",
    "NewDeviceMetadata": {
      "DeviceKey": "us-west-2_fec070d2-fa88-424a-8ec8-b26d7198eb23",
      "DeviceGroupKey": "-wt2ha1Zd"
    }
  }
}
```

- Untuk detail API, lihat [RespondToAuthChallenge](#) 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 [AWS Repositori Contoh Kode](#).

```
const respondToAuthChallenge = ({
  clientId,
  username,
  session,
  userPoolId,
  code,
}) => {
  const client = new CognitoIdentityProviderClient({});

  const command = new RespondToAuthChallengeCommand({
    ChallengeName: ChallengeNameType.SOFTWARE_TOKEN_MFA,
    ChallengeResponses: {
      SOFTWARE_TOKEN_MFA_CODE: code,
      USERNAME: username,
    },
    ClientId: clientId,
    UserPoolId: userPoolId,
    Session: session,
  });

  return client.send(command);
};
```

- Untuk detail API, lihat [RespondToAuthChallenge](#) di Referensi AWS SDK for JavaScript API.



## Python

### SDK untuk Python (Boto3)

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

Masuk dengan perangkat yang dilacak. Untuk menyelesaikan proses masuk, klien harus merespons tantangan Secure Remote Password (SRP) dengan benar.

```
class CognitoIdentityProviderWrapper:
    """Encapsulates Amazon Cognito actions"""

    def __init__(self, cognito_idp_client, user_pool_id, client_id,
                 client_secret=None):
        """
        :param cognito_idp_client: A Boto3 Amazon Cognito Identity Provider
        client.
        :param user_pool_id: The ID of an existing Amazon Cognito user pool.
        :param client_id: The ID of a client application registered with the user
        pool.
        :param client_secret: The client secret, if the client has a secret.
        """
        self.cognito_idp_client = cognito_idp_client
        self.user_pool_id = user_pool_id
        self.client_id = client_id
        self.client_secret = client_secret

    def sign_in_with_tracked_device(
        self,
        user_name,
        password,
        device_key,
        device_group_key,
        device_password,
        aws_srp,
    ):
        """
```

Signs in to Amazon Cognito as a user who has a tracked device. Signing in with a tracked device lets a user sign in without entering a new MFA code.

Signing in with a tracked device requires that the client respond to the SRP protocol. The scenario associated with this example uses the warrant package to help with SRP calculations.

For more information on SRP, see [https://en.wikipedia.org/wiki/Secure\\_Remote\\_Password\\_protocol](https://en.wikipedia.org/wiki/Secure_Remote_Password_protocol).

```
:param user_name: The user that is associated with the device.
:param password: The user's password.
:param device_key: The key of a tracked device.
:param device_group_key: The group key of a tracked device.
:param device_password: The password that is associated with the device.
:param aws_srp: A class that helps with SRP calculations. The scenario
                associated with this example uses the warrant package.
:return: The result of the authentication. When successful, this contains
an
        access token for the user.
"""
try:
    srp_helper = aws_srp.AWSSRP(
        username=user_name,
        password=device_password,
        pool_id="_",
        client_id=self.client_id,
        client_secret=None,
        client=self.cognito_idp_client,
    )

    response_init = self.cognito_idp_client.initiate_auth(
        ClientId=self.client_id,
        AuthFlow="USER_PASSWORD_AUTH",
        AuthParameters={
            "USERNAME": user_name,
            "PASSWORD": password,
            "DEVICE_KEY": device_key,
        },
    )
    if response_init["ChallengeName"] != "DEVICE_SRP_AUTH":
```

```
        raise RuntimeError(
            f"Expected DEVICE_SRP_AUTH challenge but got
{response_init['ChallengeName']})."
        )

    auth_params = srp_helper.get_auth_params()
    auth_params["DEVICE_KEY"] = device_key
    response_auth = self.cognito_idp_client.respond_to_auth_challenge(
        ClientId=self.client_id,
        ChallengeName="DEVICE_SRP_AUTH",
        ChallengeResponses=auth_params,
    )
    if response_auth["ChallengeName"] != "DEVICE_PASSWORD_VERIFIER":
        raise RuntimeError(
            f"Expected DEVICE_PASSWORD_VERIFIER challenge but got "
            f"{response_init['ChallengeName']})."
        )

    challenge_params = response_auth["ChallengeParameters"]
    challenge_params["USER_ID_FOR_SRP"] = device_group_key + device_key
    cr = srp_helper.process_challenge(challenge_params, {"USERNAME":
user_name})
    cr["USERNAME"] = user_name
    cr["DEVICE_KEY"] = device_key
    response_verifier =
self.cognito_idp_client.respond_to_auth_challenge(
        ClientId=self.client_id,
        ChallengeName="DEVICE_PASSWORD_VERIFIER",
        ChallengeResponses=cr,
    )
    auth_tokens = response_verifier["AuthenticationResult"]
except ClientError as err:
    logger.error(
        "Couldn't start client sign in for %s. Here's why: %s: %s",
        user_name,
        err.response["Error"]["Code"],
        err.response["Error"]["Message"],
    )
    raise
else:
    return auth_tokens
```

- Untuk detail API, lihat [RespondToAuthChallenge](#) di AWS SDK for Python (Boto3) Referensi API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Gunakan **SignUp** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan `SignUp`.

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:

- [Secara otomatis mengonfirmasi pengguna yang dikenal dengan fungsi Lambda](#)
- [Secara otomatis memigrasikan pengguna yang dikenal dengan fungsi Lambda](#)
- [Mendaftar pengguna dengan kumpulan pengguna yang membutuhkan MFA](#)

## .NET

### AWS SDK for .NET

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
/// <summary>
/// Sign up a new user.
/// </summary>
/// <param name="clientId">The client Id of the application.</param>
/// <param name="userName">The username to use.</param>
/// <param name="password">The user's password.</param>
/// <param name="email">The email address of the user.</param>
/// <returns>A Boolean value indicating whether the user was confirmed.</
returns>
    public async Task<bool> SignUpAsync(string clientId, string userName, string
password, string email)
```

```
{
    var userAttrs = new AttributeType
    {
        Name = "email",
        Value = email,
    };

    var userAttrsList = new List<AttributeType>();

    userAttrsList.Add(userAttrs);

    var signUpRequest = new SignUpRequest
    {
        UserAttributes = userAttrsList,
        Username = userName,
        ClientId = clientId,
        Password = password
    };

    var response = await _cognitoService.SignUpAsync(signUpRequest);
    return response.HttpStatusCode == HttpStatusCode.OK;
}
```

- Untuk detail API, lihat [SignUp](#) di Referensi AWS SDK for .NET API.

## C++

### SDK untuk C++

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
Aws::Client::ClientConfiguration clientConfig;
// Optional: Set to the AWS Region (overrides config file).
// clientConfig.region = "us-east-1";
```

```

    Aws::CognitoIdentityProvider::CognitoIdentityProviderClient
    client(clientConfig);

    Aws::CognitoIdentityProvider::Model::SignUpRequest request;
    request.AddUserAttributes(
        Aws::CognitoIdentityProvider::Model::AttributeType().WithName(
            "email").WithValue(email));
    request.SetUsername(userName);
    request.SetPassword(password);
    request.SetClientId(clientID);
    Aws::CognitoIdentityProvider::Model::SignUpOutcome outcome =
        client.SignUp(request);

    if (outcome.IsSuccess()) {
        std::cout << "The signup request for " << userName << " was
successful."
                << std::endl;
    }
    else if (outcome.GetError().GetErrorType() ==
Aws::CognitoIdentityProvider::CognitoIdentityProviderErrors::USERNAME_EXISTS) {
        std::cout
            << "The username already exists. Please enter a different
username."
            << std::endl;
        userExists = true;
    }
    else {
        std::cerr << "Error with CognitoIdentityProvider::SignUpRequest. "
                << outcome.GetError().GetMessage()
                << std::endl;
        return false;
    }
}

```

- Untuk detail API, lihat [SignUp](#) di Referensi AWS SDK for C++ API.

## CLI

### AWS CLI

Untuk mendaftar pengguna

Contoh ini mendaftar jane@example.com.

Perintah:

```
aws cognito-idp sign-up --client-id 3n4b5urk1ft4f13mg5e62d9ado --
username jane@example.com --password PASSWORD --user-attributes
  Name="email",Value="jane@example.com" Name="name",Value="Jane"
```

Output:

```
{
  "UserConfirmed": false,
  "UserSub": "e04d60a6-45dc-441c-a40b-e25a787d4862"
}
```

- Untuk detail API, lihat [SignUp](#) di Referensi AWS CLI Perintah.

Go

SDK untuk Go V2

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
type CognitoActions struct {
  CognitoClient *cognitoidentityprovider.Client
}

// SignUp signs up a user with Amazon Cognito.
func (actor CognitoActions) SignUp(clientId string, userName string, password
string, userEmail string) (bool, error) {
  confirmed := false
  output, err := actor.CognitoClient.SignUp(context.TODO(),
&cognitoidentityprovider.SignUpInput{
```

```
ClientId: aws.String(clientId),
Password: aws.String(password),
Username: aws.String(userName),
UserAttributes: []types.AttributeType{
    {Name: aws.String("email"), Value: aws.String(userEmail)},
},
})
if err != nil {
    var invalidPassword *types.InvalidPasswordException
    if errors.As(err, &invalidPassword) {
        log.Println(*invalidPassword.Message)
    } else {
        log.Printf("Couldn't sign up user %v. Here's why: %v\n", userName, err)
    }
} else {
    confirmed = output.UserConfirmed
}
return confirmed, err
}
```

- Untuk detail API, lihat [SignUp](#) di Referensi AWS SDK for Go API.

## Java

### SDK untuk Java 2.x

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
public static void signUp(CognitoIdentityProviderClient
identityProviderClient, String clientId, String userName,
    String password, String email) {
    AttributeType userAttrs = AttributeType.builder()
        .name("email")
        .value(email)
        .build();
```



```
List<AttributeType> userAttrsList = new ArrayList<>();
userAttrsList.add(userAttrs);
try {
    SignUpRequest signUpRequest = SignUpRequest.builder()
        .userAttributes(userAttrsList)
        .username(userName)
        .clientId(clientId)
        .password(password)
        .build();

    identityProviderClient.signUp(signUpRequest);
    System.out.println("User has been signed up ");

} catch (CognitoIdentityProviderException e) {
    System.err.println(e.awsErrorDetails().errorMessage());
    System.exit(1);
}
}
```

- Untuk detail API, lihat [SignUp](#) 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 [AWS Repositori Contoh Kode](#).

```
const signUp = ({ clientId, username, password, email }) => {
    const client = new CognitoIdentityProviderClient({});

    const command = new SignUpCommand({
        ClientId: clientId,
        Username: username,
        Password: password,
        UserAttributes: [{ Name: "email", Value: email }],
    });
};
```

```
return client.send(command);
};
```

- Untuk detail API, lihat [SignUp](#) di Referensi AWS SDK for JavaScript API.

## Kotlin

### SDK untuk Kotlin

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
suspend fun signUp(clientIdVal: String?, userNameVal: String?, passwordVal:
String?, emailVal: String?) {
    val userAttrs = AttributeType {
        name = "email"
        value = emailVal
    }

    val userAttrsList = mutableListOf<AttributeType>()
    userAttrsList.add(userAttrs)
    val signUpRequest = SignUpRequest {
        userAttributes = userAttrsList
        username = userNameVal
        clientId = clientIdVal
        password = passwordVal
    }

    CognitoIdentityProviderClient { region = "us-east-1" }.use
{ identityProviderClient ->
    identityProviderClient.signUp(signUpRequest)
    println("User has been signed up")
}
}
```

- Untuk detail API, lihat [SignUp](#) di AWS SDK untuk referensi API Kotlin.

## Python

### SDK untuk Python (Boto3)

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
class CognitoIdentityProviderWrapper:
    """Encapsulates Amazon Cognito actions"""

    def __init__(self, cognito_idp_client, user_pool_id, client_id,
                 client_secret=None):
        """
        :param cognito_idp_client: A Boto3 Amazon Cognito Identity Provider
        client.
        :param user_pool_id: The ID of an existing Amazon Cognito user pool.
        :param client_id: The ID of a client application registered with the user
        pool.
        :param client_secret: The client secret, if the client has a secret.
        """
        self.cognito_idp_client = cognito_idp_client
        self.user_pool_id = user_pool_id
        self.client_id = client_id
        self.client_secret = client_secret

    def sign_up_user(self, user_name, password, user_email):
        """
        Signs up a new user with Amazon Cognito. This action prompts Amazon
        Cognito
        to send an email to the specified email address. The email contains a
        code that
        can be used to confirm the user.

        When the user already exists, the user status is checked to determine
        whether
        the user has been confirmed.

        :param user_name: The user name that identifies the new user.
```

```
:param password: The password for the new user.
:param user_email: The email address for the new user.
:return: True when the user is already confirmed with Amazon Cognito.
        Otherwise, false.
"""
try:
    kwargs = {
        "ClientId": self.client_id,
        "Username": user_name,
        "Password": password,
        "UserAttributes": [{"Name": "email", "Value": user_email}],
    }
    if self.client_secret is not None:
        kwargs["SecretHash"] = self._secret_hash(user_name)
    response = self.cognito_idp_client.sign_up(**kwargs)
    confirmed = response["UserConfirmed"]
except ClientError as err:
    if err.response["Error"]["Code"] == "UsernameExistsException":
        response = self.cognito_idp_client.admin_get_user(
            UserPoolId=self.user_pool_id, Username=user_name
        )
        logger.warning(
            "User %s exists and is %s.", user_name,
            response["UserStatus"]
        )
        confirmed = response["UserStatus"] == "CONFIRMED"
    else:
        logger.error(
            "Couldn't sign up %s. Here's why: %s: %s",
            user_name,
            err.response["Error"]["Code"],
            err.response["Error"]["Message"],
        )
        raise
return confirmed
```

- Untuk detail API, lihat [SignUp](#) di AWS SDK for Python (Boto3) Referensi API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Gunakan `UpdateUserPool` dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan `UpdateUserPool`.

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:

- [Secara otomatis mengonfirmasi pengguna yang dikenal dengan fungsi Lambda](#)
- [Secara otomatis memigrasikan pengguna yang dikenal dengan fungsi Lambda](#)
- [Menulis data aktivitas khusus dengan fungsi Lambda setelah autentikasi pengguna Amazon Cognito](#)

## CLI

### AWS CLI

Untuk memperbarui kumpulan pengguna

Contoh ini menambahkan tag ke kumpulan pengguna.

Perintah:

```
aws cognito-idp update-user-pool --user-pool-id us-west-2_aaaaaaaaa --user-pool-tags Team=Blue,Area=West
```

- Untuk detail API, lihat [UpdateUserPool](#) di Referensi AWS CLI Perintah.

## Go

### SDK untuk Go V2

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
type CognitoActions struct {
    CognitoClient *cognitoidentityprovider.Client
}

// Trigger and TriggerInfo define typed data for updating an Amazon Cognito
// trigger.
type Trigger int

const (
    PreSignUp Trigger = iota
    UserMigration
    PostAuthentication
)

type TriggerInfo struct {
    Trigger    Trigger
    HandlerArn *string
}

// UpdateTriggers adds or removes Lambda triggers for a user pool. When a trigger
// is specified with a `nil` value,
// it is removed from the user pool.
func (actor CognitoActions) UpdateTriggers(userPoolId string,
    triggers ...TriggerInfo) error {
    output, err := actor.CognitoClient.DescribeUserPool(context.TODO(),
    &cognitoidentityprovider.DescribeUserPoolInput{
        UserPoolId: aws.String(userPoolId),
    })
    if err != nil {
        log.Printf("Couldn't get info about user pool %v. Here's why: %v\n",
        userPoolId, err)
        return err
    }
    lambdaConfig := output.UserPool.LambdaConfig
    for _, trigger := range triggers {
        switch trigger.Trigger {
        case PreSignUp:
            lambdaConfig.PreSignUp = trigger.HandlerArn
        case UserMigration:
            lambdaConfig.UserMigration = trigger.HandlerArn
        }
    }
}
```

```
case PostAuthentication:
    lambdaConfig.PostAuthentication = trigger.HandlerArn
}
}
_, err = actor.CognitoClient.UpdateUserPool(context.TODO(),
&cognitoidentityprovider.UpdateUserPoolInput{
    UserPoolId:    aws.String(userPoolId),
    LambdaConfig: lambdaConfig,
})
if err != nil {
    log.Printf("Couldn't update user pool %v. Here's why: %v\n", userPoolId, err)
}
return err
}
```

- Untuk detail API, lihat [UpdateUserPool](#) di Referensi AWS SDK for Go API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Gunakan **VerifySoftwareToken** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan `VerifySoftwareToken`.

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:

- [Mendaftar pengguna dengan kumpulan pengguna yang membutuhkan MFA](#)

## .NET

### AWS SDK for .NET

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
/// <summary>
/// Verify the TOTP and register for MFA.
/// </summary>
/// <param name="session">The name of the session.</param>
/// <param name="code">The MFA code.</param>
/// <returns>The status of the software token.</returns>
public async Task<VerifySoftwareTokenResponseType>
VerifySoftwareTokenAsync(string session, string code)
{
    var tokenRequest = new VerifySoftwareTokenRequest
    {
        UserCode = code,
        Session = session,
    };

    var verifyResponse = await
_cognitoService.VerifySoftwareTokenAsync(tokenRequest);

    return verifyResponse.Status;
}
```

- Untuk detail API, lihat [VerifySoftwareToken](#) di Referensi AWS SDK for .NET API.

## C++

### SDK untuk C++

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
Aws::Client::ClientConfiguration clientConfig;
// Optional: Set to the AWS Region (overrides config file).
// clientConfig.region = "us-east-1";

Aws::CognitoIdentityProvider::CognitoIdentityProviderClient
client(clientConfig);
```



```
Aws::CognitoIdentityProvider::Model::VerifySoftwareTokenRequest request;
request.SetUserCode(userCode);
request.SetSession(session);

Aws::CognitoIdentityProvider::Model::VerifySoftwareTokenOutcome outcome =
    client.VerifySoftwareToken(request);

if (outcome.IsSuccess()) {
    std::cout << "Verification of the code was successful."
              << std::endl;
    session = outcome.GetResult().GetSession();
}
else {
    std::cerr << "Error with
CognitoIdentityProvider::VerifySoftwareToken. "
              << outcome.GetError().GetMessage()
              << std::endl;
    return false;
}
```

- Untuk detail API, lihat [VerifySoftwareToken](#) di Referensi AWS SDK for C++ API.

## Java

### SDK untuk Java 2.x

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
// Verify the TOTP and register for MFA.
public static void verifyTOTP(CognitoIdentityProviderClient
identityProviderClient, String session, String code) {
    try {
        VerifySoftwareTokenRequest tokenRequest =
VerifySoftwareTokenRequest.builder()
            .userCode(code)
            .session(session)
```

```
        .build();

        VerifySoftwareTokenResponse verifyResponse =
identityProviderClient.verifySoftwareToken(tokenRequest);
        System.out.println("The status of the token is " +
verifyResponse.statusAsString());

    } catch (CognitoIdentityProviderException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
```

- Untuk detail API, lihat [VerifySoftwareToken](#) 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 [AWS Repositori Contoh Kode](#).

```
const verifySoftwareToken = (totp) => {
    const client = new CognitoIdentityProviderClient({});

    // The 'Session' is provided in the response to 'AssociateSoftwareToken'.
    const session = process.env.SESSION;

    if (!session) {
        throw new Error(
            "Missing a valid Session. Did you run 'admin-initiate-auth'?",
        );
    }

    const command = new VerifySoftwareTokenCommand({
        Session: session,
        UserCode: totp,
    });
```

```
return client.send(command);  
};
```

- Untuk detail API, lihat [VerifySoftwareToken](#) di Referensi AWS SDK for JavaScript API.

## Kotlin

### SDK untuk Kotlin

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
// Verify the TOTP and register for MFA.  
suspend fun verifyTOTP(sessionVal: String?, codeVal: String?) {  
    val tokenRequest = VerifySoftwareTokenRequest {  
        userCode = codeVal  
        session = sessionVal  
    }  
  
    CognitoIdentityProviderClient { region = "us-east-1" }.use  
{ identityProviderClient ->  
    val verifyResponse =  
identityProviderClient.verifySoftwareToken(tokenRequest)  
        println("The status of the token is ${verifyResponse.status}")  
    }  
}
```

- Untuk detail API, lihat [VerifySoftwareToken](#) di AWS SDK untuk referensi API Kotlin.

## Python

### SDK untuk Python (Boto3)

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
class CognitoIdentityProviderWrapper:
    """Encapsulates Amazon Cognito actions"""

    def __init__(self, cognito_idp_client, user_pool_id, client_id,
                 client_secret=None):
        """
        :param cognito_idp_client: A Boto3 Amazon Cognito Identity Provider
        client.
        :param user_pool_id: The ID of an existing Amazon Cognito user pool.
        :param client_id: The ID of a client application registered with the user
        pool.
        :param client_secret: The client secret, if the client has a secret.
        """
        self.cognito_idp_client = cognito_idp_client
        self.user_pool_id = user_pool_id
        self.client_id = client_id
        self.client_secret = client_secret

    def verify_mfa(self, session, user_code):
        """
        Verify a new MFA application that is associated with a user.

        :param session: Session information returned from a previous call to
        initiate
                        authentication.
        :param user_code: A code generated by the associated MFA application.
        :return: Status that indicates whether the MFA application is verified.
        """
        try:
            response = self.cognito_idp_client.verify_software_token(
                Session=session, UserCode=user_code
```

```
    )
except ClientError as err:
    logger.error(
        "Couldn't verify MFA. Here's why: %s: %s",
        err.response["Error"]["Code"],
        err.response["Error"]["Message"],
    )
    raise
else:
    response.pop("ResponseMetadata", None)
    return response
```

- Untuk detail API, lihat [VerifySoftwareToken](#) di AWS SDK for Python (Boto3) Referensi API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Skenario untuk Penyedia Identitas Amazon Cognito menggunakan SDK AWS

Contoh kode berikut menunjukkan cara menerapkan skenario umum di Amazon Cognito Identity Provider dengan AWS SDK. Skenario ini menunjukkan cara menyelesaikan tugas tertentu dengan memanggil beberapa fungsi dalam Penyedia Identitas Amazon Cognito. Setiap skenario menyertakan tautan ke GitHub, di mana Anda dapat menemukan petunjuk tentang cara mengatur dan menjalankan kode.

### Contoh

- [Secara otomatis mengonfirmasi pengguna Amazon Cognito yang dikenal dengan fungsi Lambda menggunakan SDK AWS](#)
- [Secara otomatis memigrasikan pengguna Amazon Cognito yang dikenal dengan fungsi Lambda menggunakan SDK AWS](#)
- [Mendaftar pengguna dengan kumpulan pengguna Amazon Cognito yang memerlukan MFA menggunakan SDK AWS](#)
- [Menulis data aktivitas kustom dengan fungsi Lambda setelah autentikasi pengguna Amazon Cognito menggunakan SDK AWS](#)

## Secara otomatis mengonfirmasi pengguna Amazon Cognito yang dikenal dengan fungsi Lambda menggunakan SDK AWS

Contoh kode berikut menunjukkan cara mengonfirmasi pengguna Amazon Cognito yang diketahui secara otomatis dengan fungsi Lambda.

- Konfigurasi kumpulan pengguna untuk memanggil fungsi Lambda untuk PreSignUp pemicunya.
- Daftarkan pengguna dengan Amazon Cognito.
- Fungsi Lambda memindai tabel DynamoDB dan secara otomatis mengonfirmasi pengguna yang dikenal.
- Masuk sebagai pengguna baru, lalu bersihkan sumber daya.

Go

SDK untuk Go V2

### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkap dan pelajari cara menyiapkan dan menjalankan di [Repositori Contoh Kode AWS](#).

Jalankan skenario interaktif di penggugah/prompt perintah.

```
// AutoConfirm separates the steps of this scenario into individual functions so
// that
// they are simpler to read and understand.
type AutoConfirm struct {
    helper      IScenarioHelper
    questioner  demotools.IQuestioner
    resources   Resources
    cognitoActor *actions.CognitoActions
}

// NewAutoConfirm constructs a new auto confirm runner.
func NewAutoConfirm(sdkConfig aws.Config, questioner demotools.IQuestioner,
    helper IScenarioHelper) AutoConfirm {
    scenario := AutoConfirm{
```

```
    helper:      helper,
    questioner:  questioner,
    resources:   Resources{},
    cognitoActor: &actions.CognitoActions{CognitoClient:
cognitoidentityprovider.NewFromConfig(sdkConfig)},
}
scenario.resources.init(scenario.cognitoActor, questioner)
return scenario
}

// AddPreSignUpTrigger adds a Lambda handler as an invocation target for the
PreSignUp trigger.
func (runner *AutoConfirm) AddPreSignUpTrigger(userPoolId string, functionArn
string) {
log.Printf("Let's add a Lambda function to handle the PreSignUp trigger from
Cognito.\n" +
    "This trigger happens when a user signs up, and lets your function take action
before the main Cognito\n" +
    "sign up processing occurs.\n")
err := runner.cognitoActor.UpdateTriggers(
    userPoolId,
    actions.TriggerInfo{Trigger: actions.PreSignUp, HandlerArn:
aws.String(functionArn)})
if err != nil {
    panic(err)
}
log.Printf("Lambda function %v added to user pool %v to handle the PreSignUp
trigger.\n",
    functionArn, userPoolId)
}

// SignUpUser signs up a user from the known user table with a password you
specify.
func (runner *AutoConfirm) SignUpUser(clientId string, usersTable string)
(string, string) {
log.Println("Let's sign up a user to your Cognito user pool. When the user's
email matches an email in the\n" +
    "DynamoDB known users table, it is automatically verified and the user is
confirmed.")

knownUsers, err := runner.helper.GetKnownUsers(usersTable)
if err != nil {
    panic(err)
}
}
```

```
userChoice := runner.questioner.AskChoice("Which user do you want to use?\n",
knownUsers.UserNameList())
user := knownUsers.Users[userChoice]

var signedUp bool
var userConfirmed bool
password := runner.questioner.AskPassword("Enter a password that has at least
eight characters, uppercase, lowercase, numbers and symbols.\n"+
"(the password will not display as you type):", 8)
for !signedUp {
    log.Printf("Signing up user '%v' with email '%v' to Cognito.\n", user.UserName,
user.UserEmail)
    userConfirmed, err = runner.cognitoActor.SignUp(clientId, user.UserName,
password, user.UserEmail)
    if err != nil {
        var invalidPassword *types.InvalidPasswordException
        if errors.As(err, &invalidPassword) {
            password = runner.questioner.AskPassword("Enter another password:", 8)
        } else {
            panic(err)
        }
    } else {
        signedUp = true
    }
}
log.Printf("User %v signed up, confirmed = %v.\n", user.UserName, userConfirmed)

log.Println(strings.Repeat("-", 88))

return user.UserName, password
}

// SignInUser signs in a user.
func (runner *AutoConfirm) SignInUser(clientId string, userName string, password
string) string {
    runner.questioner.Ask("Press Enter when you're ready to continue.")
    log.Printf("Let's sign in as %v...\n", userName)
    authResult, err := runner.cognitoActor.SignIn(clientId, userName, password)
    if err != nil {
        panic(err)
    }
    log.Printf("Successfully signed in. Your access token starts with: %v...\n",
(*authResult.AccessToken)[:10])
    log.Println(strings.Repeat("-", 88))
}
```



```

    return *authResult.AccessToken
}

// Run runs the scenario.
func (runner *AutoConfirm) Run(stackName string) {
    defer func() {
        if r := recover(); r != nil {
            log.Println("Something went wrong with the demo.")
            runner.resources.Cleanup()
        }
    }()

    log.Println(strings.Repeat("-", 88))
    log.Printf("Welcome\n")

    log.Println(strings.Repeat("-", 88))

    stackOutputs, err := runner.helper.GetStackOutputs(stackName)
    if err != nil {
        panic(err)
    }
    runner.resources.userPoolId = stackOutputs["UserPoolId"]
    runner.helper.PopulateUserTable(stackOutputs["TableName"])

    runner.AddPreSignUpTrigger(stackOutputs["UserPoolId"],
        stackOutputs["AutoConfirmFunctionArn"])
    runner.resources.triggers = append(runner.resources.triggers, actions.PreSignUp)
    userName, password := runner.SignUpUser(stackOutputs["UserPoolClientId"],
        stackOutputs["TableName"])
    runner.helper.ListRecentLogEvents(stackOutputs["AutoConfirmFunction"])
    runner.resources.userAccessTokens = append(runner.resources.userAccessTokens,
        runner.SignInUser(stackOutputs["UserPoolClientId"], userName, password))

    runner.resources.Cleanup()

    log.Println(strings.Repeat("-", 88))
    log.Println("Thanks for watching!")
    log.Println(strings.Repeat("-", 88))
}

```

Tangani PreSignUp pelatuk dengan fungsi Lambda.

```
const TABLE_NAME = "TABLE_NAME"

// UserInfo defines structured user data that can be marshalled to a DynamoDB
// format.
type UserInfo struct {
    UserName string `dynamodbav:"UserName"`
    UserEmail string `dynamodbav:"UserEmail"`
}

// GetKey marshals the user email value to a DynamoDB key format.
func (user UserInfo) GetKey() map[string]dynamodbtypes.AttributeValue {
    userEmail, err := attributevalue.Marshal(user.UserEmail)
    if err != nil {
        panic(err)
    }
    return map[string]dynamodbtypes.AttributeValue{"UserEmail": userEmail}
}

type handler struct {
    dynamoClient *dynamodb.Client
}

// HandleRequest handles the PreSignUp event by looking up a user in an Amazon
// DynamoDB table and
// specifying whether they should be confirmed and verified.
func (h *handler) HandleRequest(ctx context.Context, event
events.CognitoEventUserPoolsPreSignup) (events.CognitoEventUserPoolsPreSignup,
error) {
    log.Printf("Received presignup from %v for user '%v'", event.TriggerSource,
event.UserName)
    if event.TriggerSource != "PreSignUp_SignUp" {
        // Other trigger sources, such as PreSignUp_AdminInitiateAuth, ignore the
        // response from this handler.
        return event, nil
    }
    tableName := os.Getenv(TABLE_NAME)
    user := UserInfo{
        UserEmail: event.Request.UserAttributes["email"],
    }
    log.Printf("Looking up email %v in table %v.\n", user.UserEmail, tableName)
    output, err := h.dynamoClient.GetItem(ctx, &dynamodb.GetItemInput{
        Key:      user.GetKey(),
```

```
    TableName: aws.String(tableName),
  })
  if err != nil {
    log.Printf("Error looking up email %v.\n", user.UserEmail)
    return event, err
  }
  if output.Item == nil {
    log.Printf("Email %v not found. Email verification is required.\n",
      user.UserEmail)
    return event, err
  }

  err = attributevalue.UnmarshalMap(output.Item, &user)
  if err != nil {
    log.Printf("Couldn't unmarshal DynamoDB item. Here's why: %v\n", err)
    return event, err
  }

  if user.UserName != event.UserName {
    log.Printf("UserEmail %v found, but stored UserName '%v' does not match
      supplied UserName '%v'. Verification is required.\n",
      user.UserEmail, user.UserName, event.UserName)
  } else {
    log.Printf("UserEmail %v found with matching UserName %v. User is confirmed.
\n", user.UserEmail, user.UserName)
    event.Response.AutoConfirmUser = true
    event.Response.AutoVerifyEmail = true
  }

  return event, err
}

func main() {
  sdkConfig, err := config.LoadDefaultConfig(context.TODO())
  if err != nil {
    log.Panicln(err)
  }
  h := handler{
    dynamoClient: dynamodb.NewFromConfig(sdkConfig),
  }
  lambda.Start(h.HandleRequest)
}
```

Buat struct yang melakukan tugas-tugas umum.

```
// IScenarioHelper defines common functions used by the workflows in this
// example.
type IScenarioHelper interface {
    Pause(secs int)
    GetStackOutputs(stackName string) (actions.StackOutputs, error)
    PopulateUserTable(tableName string)
    GetKnownUsers(tableName string) (actions.UserList, error)
    AddKnownUser(tableName string, user actions.User)
    ListRecentLogEvents(functionName string)
}

// ScenarioHelper contains AWS wrapper structs used by the workflows in this
// example.
type ScenarioHelper struct {
    questioner demotools.IQuestioner
    dynamoActor *actions.DynamoActions
    cfnActor *actions.CloudFormationActions
    cwActor *actions.CloudWatchLogsActions
    isTestRun bool
}

// NewScenarioHelper constructs a new scenario helper.
func NewScenarioHelper(sdkConfig aws.Config, questioner demotools.IQuestioner)
ScenarioHelper {
    scenario := ScenarioHelper{
        questioner: questioner,
        dynamoActor: &actions.DynamoActions{DynamoClient:
        dynamodb.NewFromConfig(sdkConfig)},
        cfnActor: &actions.CloudFormationActions{CfnClient:
        cloudformation.NewFromConfig(sdkConfig)},
        cwActor: &actions.CloudWatchLogsActions{CwlClient:
        cloudwatchlogs.NewFromConfig(sdkConfig)},
    }
    return scenario
}

// Pause waits for the specified number of seconds.
func (helper ScenarioHelper) Pause(secs int) {
```

```
    if !helper.isTestRun {
        time.Sleep(time.Duration(secs) * time.Second)
    }
}

// GetStackOutputs gets the outputs from the specified CloudFormation stack in a
// structured format.
func (helper ScenarioHelper) GetStackOutputs(stackName string)
(actions.StackOutputs, error) {
    return helper.cfnActor.GetOutputs(stackName), nil
}

// PopulateUserTable fills the known user table with example data.
func (helper ScenarioHelper) PopulateUserTable(tableName string) {
    log.Printf("First, let's add some users to the DynamoDB %v table we'll use for
this example.\n", tableName)
    err := helper.dynamoActor.PopulateTable(tableName)
    if err != nil {
        panic(err)
    }
}

// GetKnownUsers gets the users from the known users table in a structured
// format.
func (helper ScenarioHelper) GetKnownUsers(tableName string) (actions.UserList,
error) {
    knownUsers, err := helper.dynamoActor.Scan(tableName)
    if err != nil {
        log.Printf("Couldn't get known users from table %v. Here's why: %v\n",
tableName, err)
    }
    return knownUsers, err
}

// AddKnownUser adds a user to the known users table.
func (helper ScenarioHelper) AddKnownUser(tableName string, user actions.User) {
    log.Printf("Adding user '%v' with email '%v' to the DynamoDB known users
table...\n",
user.UserName, user.UserEmail)
    err := helper.dynamoActor.AddUser(tableName, user)
    if err != nil {
        panic(err)
    }
}
```

```
// ListRecentLogEvents gets the most recent log stream and events for the
// specified Lambda function and displays them.
func (helper ScenarioHelper) ListRecentLogEvents(functionName string) {
    log.Println("Waiting a few seconds to let Lambda write to CloudWatch Logs...")
    helper.Pause(10)
    log.Println("Okay, let's check the logs to find what's happened recently with
    your Lambda function.")
    logStream, err := helper.cwlActor.GetLatestLogStream(functionName)
    if err != nil {
        panic(err)
    }
    log.Printf("Getting some recent events from log stream %v\n",
    *logStream.LogStreamName)
    events, err := helper.cwlActor.GetLogEvents(functionName,
    *logStream.LogStreamName, 10)
    if err != nil {
        panic(err)
    }
    for _, event := range events {
        log.Printf("\t\t%v", *event.Message)
    }
    log.Println(strings.Repeat("-", 88))
}
```

Buat struct yang membungkus tindakan Amazon Cognito.

```
type CognitoActions struct {
    CognitoClient *cognitoidentityprovider.Client
}

// Trigger and TriggerInfo define typed data for updating an Amazon Cognito
// trigger.
type Trigger int

const (
    PreSignUp Trigger = iota
```

```
UserMigration
PostAuthentication
)

type TriggerInfo struct {
    Trigger    Trigger
    HandlerArn *string
}

// UpdateTriggers adds or removes Lambda triggers for a user pool. When a trigger
// is specified with a `nil` value,
// it is removed from the user pool.
func (actor CognitoActions) UpdateTriggers(userPoolId string,
    triggers ...TriggerInfo) error {
    output, err := actor.CognitoClient.DescribeUserPool(context.TODO(),
        &cognitoidentityprovider.DescribeUserPoolInput{
            UserPoolId: aws.String(userPoolId),
        })
    if err != nil {
        log.Printf("Couldn't get info about user pool %v. Here's why: %v\n",
            userPoolId, err)
        return err
    }
    lambdaConfig := output.UserPool.LambdaConfig
    for _, trigger := range triggers {
        switch trigger.Trigger {
        case PreSignUp:
            lambdaConfig.PreSignUp = trigger.HandlerArn
        case UserMigration:
            lambdaConfig.UserMigration = trigger.HandlerArn
        case PostAuthentication:
            lambdaConfig.PostAuthentication = trigger.HandlerArn
        }
    }
    _, err = actor.CognitoClient.UpdateUserPool(context.TODO(),
        &cognitoidentityprovider.UpdateUserPoolInput{
            UserPoolId:    aws.String(userPoolId),
            LambdaConfig: lambdaConfig,
        })
    if err != nil {
        log.Printf("Couldn't update user pool %v. Here's why: %v\n", userPoolId, err)
    }
    return err
}
```

```
// SignUp signs up a user with Amazon Cognito.
func (actor CognitoActions) SignUp(clientId string, userName string, password
string, userEmail string) (bool, error) {
    confirmed := false
    output, err := actor.CognitoClient.SignUp(context.TODO(),
    &cognitoidentityprovider.SignUpInput{
        ClientId: aws.String(clientId),
        Password: aws.String(password),
        Username: aws.String(userName),
        UserAttributes: []types.AttributeType{
            {Name: aws.String("email"), Value: aws.String(userEmail)},
        },
    })
    if err != nil {
        var invalidPassword *types.InvalidPasswordException
        if errors.As(err, &invalidPassword) {
            log.Println(*invalidPassword.Message)
        } else {
            log.Printf("Couldn't sign up user %v. Here's why: %v\n", userName, err)
        }
    } else {
        confirmed = output.UserConfirmed
    }
    return confirmed, err
}

// SignIn signs in a user to Amazon Cognito using a username and password
authentication flow.
func (actor CognitoActions) SignIn(clientId string, userName string, password
string) (*types.AuthenticationResultType, error) {
    var authResult *types.AuthenticationResultType
    output, err := actor.CognitoClient.InitiateAuth(context.TODO(),
    &cognitoidentityprovider.InitiateAuthInput{
        AuthFlow:      "USER_PASSWORD_AUTH",
        ClientId:      aws.String(clientId),
        AuthParameters: map[string]string{"USERNAME": userName, "PASSWORD": password},
    })
    if err != nil {
        var resetRequired *types.PasswordResetRequiredException
```



```
    if errors.As(err, &resetRequired) {
        log.Println(*resetRequired.Message)
    } else {
        log.Printf("Couldn't sign in user %v. Here's why: %v\n", userName, err)
    }
} else {
    authResult = output.AuthenticationResult
}
return authResult, err
}

// ForgotPassword starts a password recovery flow for a user. This flow typically
// sends a confirmation code
// to the user's configured notification destination, such as email.
func (actor CognitoActions) ForgotPassword(clientId string, userName string)
(*types.CodeDeliveryDetailsType, error) {
    output, err := actor.CognitoClient.ForgotPassword(context.TODO(),
&cognitoidentityprovider.ForgotPasswordInput{
    ClientId: aws.String(clientId),
    Username: aws.String(userName),
})
    if err != nil {
        log.Printf("Couldn't start password reset for user '%v'. Here;s why: %v\n",
userName, err)
    }
    return output.CodeDeliveryDetails, err
}

// ConfirmForgotPassword confirms a user with a confirmation code and a new
// password.
func (actor CognitoActions) ConfirmForgotPassword(clientId string, code string,
userName string, password string) error {
    _, err := actor.CognitoClient.ConfirmForgotPassword(context.TODO(),
&cognitoidentityprovider.ConfirmForgotPasswordInput{
    ClientId:      aws.String(clientId),
    ConfirmationCode: aws.String(code),
    Password:      aws.String(password),
    Username:      aws.String(userName),
})
    if err != nil {
```

```
var invalidPassword *types.InvalidPasswordException
if errors.As(err, &invalidPassword) {
    log.Println(*invalidPassword.Message)
} else {
    log.Printf("Couldn't confirm user %v. Here's why: %v", userName, err)
}
}
return err
}

// DeleteUser removes a user from the user pool.
func (actor CognitoActions) DeleteUser(userAccessToken string) error {
    _, err := actor.CognitoClient.DeleteUser(context.TODO(),
        &cognitoidentityprovider.DeleteUserInput{
            AccessToken: aws.String(userAccessToken),
        })
    if err != nil {
        log.Printf("Couldn't delete user. Here's why: %v\n", err)
    }
    return err
}

// AdminCreateUser uses administrator credentials to add a user to a user pool.
// This method leaves the user
// in a state that requires they enter a new password next time they sign in.
func (actor CognitoActions) AdminCreateUser(userPoolId string, userName string,
    userEmail string) error {
    _, err := actor.CognitoClient.AdminCreateUser(context.TODO(),
        &cognitoidentityprovider.AdminCreateUserInput{
            UserPoolId:      aws.String(userPoolId),
            Username:       aws.String(userName),
            MessageAction: types.MessageActionTypeSuppress,
            UserAttributes: []types.AttributeType{{Name: aws.String("email"), Value:
                aws.String(userEmail)}}},
        })
    if err != nil {
        var userExists *types.UsernameExistsException
        if errors.As(err, &userExists) {
            log.Printf("User %v already exists in the user pool.", userName)
            err = nil
        }
    }
}
```

```

    } else {
        log.Printf("Couldn't create user %v. Here's why: %v\n", userName, err)
    }
}
return err
}

// AdminSetUserPassword uses administrator credentials to set a password for a
// user without requiring a
// temporary password.
func (actor CognitoActions) AdminSetUserPassword(userPoolId string, userName
string, password string) error {
    _, err := actor.CognitoClient.AdminSetUserPassword(context.TODO(),
&cognitoidentityprovider.AdminSetUserPasswordInput{
    Password:    aws.String(password),
    UserPoolId:  aws.String(userPoolId),
    Username:    aws.String(userName),
    Permanent:   true,
})
    if err != nil {
        var invalidPassword *types.InvalidPasswordException
        if errors.As(err, &invalidPassword) {
            log.Println(*invalidPassword.Message)
        } else {
            log.Printf("Couldn't set password for user %v. Here's why: %v\n", userName,
err)
        }
    }
    return err
}

```

Buat struct yang membungkus tindakan DynamoDB.

```

// DynamoActions encapsulates the Amazon Simple Notification Service (Amazon SNS)
// actions
// used in the examples.
type DynamoActions struct {
    DynamoClient *dynamodb.Client
}

```

```
}

// User defines structured user data.
type User struct {
    UserName string
    UserEmail string
    LastLogin *LoginInfo `dynamodbav:",omitempty"`
}

// LoginInfo defines structured custom login data.
type LoginInfo struct {
    UserPoolId string
    ClientId string
    Time string
}

// UserList defines a list of users.
type UserList struct {
    Users []User
}

// UserNameList returns the usernames contained in a UserList as a list of
strings.
func (users *UserList) UserNameList() []string {
    names := make([]string, len(users.Users))
    for i := 0; i < len(users.Users); i++ {
        names[i] = users.Users[i].UserName
    }
    return names
}

// PopulateTable adds a set of test users to the table.
func (actor DynamoActions) PopulateTable(tableName string) error {
    var err error
    var item map[string]types.AttributeValue
    var writeReqs []types.WriteRequest
    for i := 1; i < 4; i++ {
        item, err = attributevalue.MarshalMap(User{UserName: fmt.Sprintf("test_user_
%v", i), UserEmail: fmt.Sprintf("test_email_%v@example.com", i)})
        if err != nil {
            log.Printf("Couldn't marshall user into DynamoDB format. Here's why: %v\n",
err)
            return err
        }
    }
}
```

```
writeReqs = append(writeReqs, types.WriteRequest{PutRequest:
&types.PutRequest{Item: item}})
}
_, err = actor.DynamoClient.BatchWriteItem(context.TODO(),
&dynamodb.BatchWriteItemInput{
RequestItems: map[string][]types.WriteRequest{tableName: writeReqs},
})
if err != nil {
log.Printf("Couldn't populate table %v with users. Here's why: %v\n",
tableName, err)
}
return err
}

// Scan scans the table for all items.
func (actor DynamoActions) Scan(tableName string) (UserList, error) {
var userList UserList
output, err := actor.DynamoClient.Scan(context.TODO(), &dynamodb.ScanInput{
TableName: aws.String(tableName),
})
if err != nil {
log.Printf("Couldn't scan table %v for items. Here's why: %v\n", tableName,
err)
} else {
err = attributevalue.UnmarshalListOfMaps(output.Items, &userList.Users)
if err != nil {
log.Printf("Couldn't unmarshal items into users. Here's why: %v\n", err)
}
}
return userList, err
}

// AddUser adds a user item to a table.
func (actor DynamoActions) AddUser(tableName string, user User) error {
userItem, err := attributevalue.MarshalMap(user)
if err != nil {
log.Printf("Couldn't marshall user to item. Here's why: %v\n", err)
}
_, err = actor.DynamoClient.PutItem(context.TODO(), &dynamodb.PutItemInput{
Item: userItem,
TableName: aws.String(tableName),
})
if err != nil {
log.Printf("Couldn't put item in table %v. Here's why: %v", tableName, err)
}
```

```
}  
return err  
}
```

Buat struct yang membungkus tindakan CloudWatch Log.

```
type CloudWatchLogsActions struct {  
    CwlClient *cloudwatchlogs.Client  
}  
  
// GetLatestLogStream gets the most recent log stream for a Lambda function.  
func (actor CloudWatchLogsActions) GetLatestLogStream(functionName string)  
    (types.LogStream, error) {  
    var logStream types.LogStream  
    logGroupName := fmt.Sprintf("/aws/lambda/%s", functionName)  
    output, err := actor.CwlClient.DescribeLogStreams(context.TODO(),  
        &cloudwatchlogs.DescribeLogStreamsInput{  
            Descending:    aws.Bool(true),  
            Limit:         aws.Int32(1),  
            LogGroupName:  aws.String(logGroupName),  
            OrderBy:      types.OrderByLastEventTime,  
        })  
    if err != nil {  
        log.Printf("Couldn't get log streams for log group %v. Here's why: %v\n",  
            logGroupName, err)  
    } else {  
        logStream = output.LogStreams[0]  
    }  
    return logStream, err  
}  
  
// GetLogEvents gets the most recent eventCount events from the specified log  
    stream.  
func (actor CloudWatchLogsActions) GetLogEvents(functionName string,  
    logStreamName string, eventCount int32) (  
    []types.OutputLogEvent, error) {  
    var events []types.OutputLogEvent  
    logGroupName := fmt.Sprintf("/aws/lambda/%s", functionName)  
    output, err := actor.CwlClient.GetLogEvents(context.TODO(),  
        &cloudwatchlogs.GetLogEventsInput{
```

```

    LogStreamName: aws.String(logStreamName),
    Limit:         aws.Int32(eventCount),
    LogGroupName:  aws.String(logGroupName),
  })
  if err != nil {
    log.Printf("Couldn't get log event for log stream %v. Here's why: %v\n",
      logStreamName, err)
  } else {
    events = output.Events
  }
  return events, err
}

```

Buat struct yang membungkus tindakan AWS CloudFormation .

```

// StackOutputs defines a map of outputs from a specific stack.
type StackOutputs map[string]string

type CloudFormationActions struct {
  CfnClient *cloudformation.Client
}

// GetOutputs gets the outputs from a CloudFormation stack and puts them into a
// structured format.
func (actor CloudFormationActions) GetOutputs(stackName string) StackOutputs {
  output, err := actor.CfnClient.DescribeStacks(context.TODO(),
    &cloudformation.DescribeStacksInput{
      StackName: aws.String(stackName),
    })
  if err != nil || len(output.Stacks) == 0 {
    log.Panicf("Couldn't find a CloudFormation stack named %v. Here's why: %v\n",
      stackName, err)
  }
  stackOutputs := StackOutputs{}
  for _, out := range output.Stacks[0].Outputs {
    stackOutputs[*out.OutputKey] = *out.OutputValue
  }
  return stackOutputs
}

```

## Pembersihan sumber daya

```
// Resources keeps track of AWS resources created during an example and handles
// cleanup when the example finishes.
type Resources struct {
    userPoolId      string
    userAccessTokens []string
    triggers        []actions.Trigger

    cognitoActor *actions.CognitoActions
    questioner   demotools.IQuestioner
}

func (resources *Resources) init(cognitoActor *actions.CognitoActions, questioner
demotools.IQuestioner) {
    resources.userAccessTokens = []string{}
    resources.triggers = []actions.Trigger{}
    resources.cognitoActor = cognitoActor
    resources.questioner = questioner
}

// Cleanup deletes all AWS resources created during an example.
func (resources *Resources) Cleanup() {
    defer func() {
        if r := recover(); r != nil {
            log.Printf("Something went wrong during cleanup.\n%v\n", r)
            log.Println("Use the AWS Management Console to remove any remaining resources
\n" +
                "that were created for this scenario.")
        }
    }()

    wantDelete := resources.questioner.AskBool("Do you want to remove all of the AWS
resources that were created "+
    "during this demo (y/n)?", "y")
    if wantDelete {
        for _, accessToken := range resources.userAccessTokens {
            err := resources.cognitoActor.DeleteUser(accessToken)
            if err != nil {
                log.Println("Couldn't delete user during cleanup.")
            }
        }
    }
}
```



```
    panic(err)
  }
  log.Println("Deleted user.")
}
triggerList := make([]actions.TriggerInfo, len(resources.triggers))
for i := 0; i < len(resources.triggers); i++ {
    triggerList[i] = actions.TriggerInfo{Trigger: resources.triggers[i],
HandlerArn: nil}
}
err := resources.cognitoActor.UpdateTriggers(resources.userPoolId,
triggerList...)
if err != nil {
    log.Println("Couldn't update Cognito triggers during cleanup.")
    panic(err)
}
log.Println("Removed Cognito triggers from user pool.")
} else {
    log.Println("Be sure to remove resources when you're done with them to avoid
unexpected charges!")
}
}
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK for Go .
  - [DeleteUser](#)
  - [InitiateAuth](#)
  - [SignUp](#)
  - [UpdateUserPool](#)

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.


## Secara otomatis memigrasikan pengguna Amazon Cognito yang dikenal dengan fungsi Lambda menggunakan SDK AWS

Contoh kode berikut menunjukkan cara memigrasi pengguna Amazon Cognito yang dikenal secara otomatis dengan fungsi Lambda.

- Konfigurasi kumpulan pengguna untuk memanggil fungsi Lambda untuk `MigrateUser` pemicunya.
- Masuk ke Amazon Cognito dengan nama pengguna dan email yang tidak ada di kumpulan pengguna.
- Fungsi Lambda memindai tabel DynamoDB dan secara otomatis memigrasikan pengguna yang dikenal ke kumpulan pengguna.
- Lakukan alur lupa kata sandi untuk mengatur ulang kata sandi untuk pengguna yang dimigrasi.
- Masuk sebagai pengguna baru, lalu bersihkan sumber daya.

Go

SDK untuk Go V2

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkap dan pelajari cara menyiapkan dan menjalankan di [Repositori Contoh Kode AWS](#).

Jalankan skenario interaktif di penggugah/prompt perintah.

```
import (
    "errors"
    "fmt"
    "log"
    "strings"
    "user_pools_and_lambda_triggers/actions"

    "github.com/aws/aws-sdk-go-v2/aws"
    "github.com/aws/aws-sdk-go-v2/service/cognitoidentityprovider"
    "github.com/aws/aws-sdk-go-v2/service/cognitoidentityprovider/types"
    "github.com/awsdocs/aws-doc-sdk-examples/gov2/demotools"
)

// MigrateUser separates the steps of this scenario into individual functions so
// that
// they are simpler to read and understand.
type MigrateUser struct {
    helper      IScenarioHelper
```

```
questioner demotools.IQuestioner
resources Resources
cognitoActor *actions.CognitoActions
}

// NewMigrateUser constructs a new migrate user runner.
func NewMigrateUser(sdkConfig aws.Config, questioner demotools.IQuestioner,
helper IScenarioHelper) MigrateUser {
scenario := MigrateUser{
helper: helper,
questioner: questioner,
resources: Resources{},
cognitoActor: &actions.CognitoActions{CognitoClient:
cognitoidentityprovider.NewFromConfig(sdkConfig)},
}
scenario.resources.init(scenario.cognitoActor, questioner)
return scenario
}

// AddMigrateUserTrigger adds a Lambda handler as an invocation target for the
MigrateUser trigger.
func (runner *MigrateUser) AddMigrateUserTrigger(userPoolId string, functionArn
string) {
log.Printf("Let's add a Lambda function to handle the MigrateUser trigger from
Cognito.\n" +
"This trigger happens when an unknown user signs in, and lets your function
take action before Cognito\n" +
"rejects the user.\n\n")
err := runner.cognitoActor.UpdateTriggers(
userPoolId,
actions.TriggerInfo{Trigger: actions.UserMigration, HandlerArn:
aws.String(functionArn)})
if err != nil {
panic(err)
}
log.Printf("Lambda function %v added to user pool %v to handle the MigrateUser
trigger.\n",
functionArn, userPoolId)

log.Println(strings.Repeat("-", 88))
}

// SignInUser adds a new user to the known users table and signs that user in to
Amazon Cognito.
```

```
func (runner *MigrateUser) SignInUser(usersTable string, clientId string) (bool,
actions.User) {
    log.Println("Let's sign in a user to your Cognito user pool. When the username
and email matches an entry in the\n" +
        "DynamoDB known users table, the email is automatically verified and the user
is migrated to the Cognito user pool.")

    user := actions.User{}
    user.UserName = runner.questioner.Ask("\nEnter a username:")
    user.UserEmail = runner.questioner.Ask("\nEnter an email that you own. This
email will be used to confirm user migration\n" +
        "during this example:")

    runner.helper.AddKnownUser(usersTable, user)

    var err error
    var resetRequired *types.PasswordResetRequiredException
    var authResult *types.AuthenticationResultType
    signedIn := false
    for !signedIn && resetRequired == nil {
        log.Printf("Signing in to Cognito as user '%v'. The expected result is a
PasswordResetRequiredException.\n\n", user.UserName)
        authResult, err = runner.cognitoActor.SignIn(clientId, user.UserName, "_")
        if err != nil {
            if errors.As(err, &resetRequired) {
                log.Printf("\nUser '%v' is not in the Cognito user pool but was found in the
DynamoDB known users table.\n"+
                    "User migration is started and a password reset is required.",
                    user.UserName)
            } else {
                panic(err)
            }
        } else {
            log.Printf("User '%v' successfully signed in. This is unexpected and probably
means you have not\n"+
                "cleaned up a previous run of this scenario, so the user exist in the Cognito
user pool.\n"+
                "You can continue this example and select to clean up resources, or manually
remove\n"+
                "the user from your user pool and try again.", user.UserName)
            runner.resources.userAccessTokens = append(runner.resources.userAccessTokens,
                *authResult.AccessToken)
            signedIn = true
        }
    }
}
```

```
}

log.Println(strings.Repeat("-", 88))
return resetRequired != nil, user
}

// ResetPassword starts a password recovery flow.
func (runner *MigrateUser) ResetPassword(clientId string, user actions.User) {
    wantCode := runner.questioner.AskBool(fmt.Sprintf("In order to migrate the user
to Cognito, you must be able to receive a confirmation\n"+
    "code by email at %v. Do you want to send a code (y/n)?", user.UserEmail), "y")
    if !wantCode {
        log.Println("To complete this example and successfully migrate a user to
Cognito, you must enter an email\n" +
        "you own that can receive a confirmation code.")
        return
    }
    codeDelivery, err := runner.cognitoActor.ForgotPassword(clientId, user.UserName)
    if err != nil {
        panic(err)
    }
    log.Printf("\nA confirmation code has been sent to %v.",
    *codeDelivery.Destination)
    code := runner.questioner.Ask("Check your email and enter it here:")

    confirmed := false
    password := runner.questioner.AskPassword("\nEnter a password that has at least
eight characters, uppercase, lowercase, numbers and symbols.\n"+
    "(the password will not display as you type):", 8)
    for !confirmed {
        log.Printf("\nConfirming password reset for user '%v'.\n", user.UserName)
        err = runner.cognitoActor.ConfirmForgotPassword(clientId, code, user.UserName,
password)
        if err != nil {
            var invalidPassword *types.InvalidPasswordException
            if errors.As(err, &invalidPassword) {
                password = runner.questioner.AskPassword("\nEnter another password:", 8)
            } else {
                panic(err)
            }
        } else {
            confirmed = true
        }
    }
}
```

```
log.Printf("User '%v' successfully confirmed and migrated.\n", user.UserName)
log.Println("Signing in with your username and password...")
authResult, err := runner.cognitoActor.SignIn(clientId, user.UserName, password)
if err != nil {
    panic(err)
}
log.Printf("Successfully signed in. Your access token starts with: %v...\n",
(*authResult.AccessToken)[:10])
runner.resources.userAccessTokens = append(runner.resources.userAccessTokens,
*authResult.AccessToken)

log.Println(strings.Repeat("-", 88))
}

// Run runs the scenario.
func (runner *MigrateUser) Run(stackName string) {
    defer func() {
        if r := recover(); r != nil {
            log.Println("Something went wrong with the demo.")
            runner.resources.Cleanup()
        }
    }()

    log.Println(strings.Repeat("-", 88))
    log.Printf("Welcome\n")

    log.Println(strings.Repeat("-", 88))

    stackOutputs, err := runner.helper.GetStackOutputs(stackName)
    if err != nil {
        panic(err)
    }
    runner.resources.userPoolId = stackOutputs["UserPoolId"]

    runner.AddMigrateUserTrigger(stackOutputs["UserPoolId"],
stackOutputs["MigrateUserFunctionArn"])
    runner.resources.triggers = append(runner.resources.triggers,
actions.UserMigration)
    resetNeeded, user := runner.SignInUser(stackOutputs["TableName"],
stackOutputs["UserPoolClientId"])
    if resetNeeded {
        runner.helper.ListRecentLogEvents(stackOutputs["MigrateUserFunction"])
        runner.ResetPassword(stackOutputs["UserPoolClientId"], user)
    }
}
```

```
runner.resources.Cleanup()

log.Println(strings.Repeat("-", 88))
log.Println("Thanks for watching!")
log.Println(strings.Repeat("-", 88))
}
```

Tangani `MigrateUser` pelatuk dengan fungsi Lambda.

```
const TABLE_NAME = "TABLE_NAME"

// UserInfo defines structured user data that can be marshalled to a DynamoDB
// format.
type UserInfo struct {
    UserName string `dynamodbav:"UserName"`
    UserEmail string `dynamodbav:"UserEmail"`
}

type handler struct {
    dynamoClient *dynamodb.Client
}

// HandleRequest handles the MigrateUser event by looking up a user in an Amazon
// DynamoDB table and
// specifying whether they should be migrated to the user pool.
func (h *handler) HandleRequest(ctx context.Context, event
events.CognitoEventUserPoolsMigrateUser)
(events.CognitoEventUserPoolsMigrateUser, error) {
    log.Printf("Received migrate trigger from %v for user '%v'",
event.TriggerSource, event.UserName)
    if event.TriggerSource != "UserMigration_Authentication" {
        return event, nil
    }
    tableName := os.Getenv(TABLE_NAME)
    user := UserInfo{
        UserName: event.UserName,
    }
    log.Printf("Looking up user '%v' in table %v.\n", user.UserName, tableName)
    filterEx := expression.Name("UserName").Equal(expression.Value(user.UserName))
```

```
expr, err := expression.NewBuilder().WithFilter(filterEx).Build()
if err != nil {
    log.Printf("Error building expression to query for user '%v'.\n",
user.UserName)
    return event, err
}
output, err := h.dynamoClient.Scan(ctx, &dynamodb.ScanInput{
    TableName:          aws.String(tableName),
    FilterExpression:   expr.Filter(),
    ExpressionAttributeNames: expr.Names(),
    ExpressionAttributeValues: expr.Values(),
})
if err != nil {
    log.Printf("Error looking up user '%v'.\n", user.UserName)
    return event, err
}
if output.Items == nil || len(output.Items) == 0 {
    log.Printf("User '%v' not found, not migrating user.\n", user.UserName)
    return event, err
}

var users []UserInfo
err = attributevalue.UnmarshalListOfMaps(output.Items, &users)
if err != nil {
    log.Printf("Couldn't unmarshal DynamoDB items. Here's why: %v\n", err)
    return event, err
}

user = users[0]
log.Printf("UserName '%v' found with email %v. User is migrated and must reset
password.\n", user.UserName, user.UserEmail)
event.CognitoEventUserPoolsMigrateUserResponse.UserAttributes =
map[string]string{
    "email":          user.UserEmail,
    "email_verified": "true", // email_verified is required for the forgot password
flow.
}
event.CognitoEventUserPoolsMigrateUserResponse.FinalUserStatus =
"RESET_REQUIRED"
event.CognitoEventUserPoolsMigrateUserResponse.MessageAction = "SUPPRESS"

return event, err
}
```



```
func main() {
    sdkConfig, err := config.LoadDefaultConfig(context.TODO())
    if err != nil {
        log.Panicln(err)
    }
    h := handler{
        dynamoClient: dynamodb.NewFromConfig(sdkConfig),
    }
    lambda.Start(h.HandleRequest)
}
```

Buat struct yang melakukan tugas-tugas umum.

```
// IScenarioHelper defines common functions used by the workflows in this
// example.
type IScenarioHelper interface {
    Pause(secs int)
    GetStackOutputs(stackName string) (actions.StackOutputs, error)
    PopulateUserTable(tableName string)
    GetKnownUsers(tableName string) (actions.UserList, error)
    AddKnownUser(tableName string, user actions.User)
    ListRecentLogEvents(functionName string)
}

// ScenarioHelper contains AWS wrapper structs used by the workflows in this
// example.
type ScenarioHelper struct {
    questioner demotools.IQuestioner
    dynamoActor *actions.DynamoActions
    cfnActor *actions.CloudFormationActions
    cwActor *actions.CloudWatchLogsActions
    isTestRun bool
}

// NewScenarioHelper constructs a new scenario helper.
func NewScenarioHelper(sdkConfig aws.Config, questioner demotools.IQuestioner)
ScenarioHelper {
    scenario := ScenarioHelper{
        questioner: questioner,
```

```
dynamoActor: &actions.DynamoActions{DynamoClient:
dynamodb.NewFromConfig(sdkConfig)},
cfnActor:    &actions.CloudFormationActions{CfnClient:
cloudformation.NewFromConfig(sdkConfig)},
cwlActor:    &actions.CloudWatchLogsActions{CwlClient:
cloudwatchlogs.NewFromConfig(sdkConfig)},
}
return scenario
}

// Pause waits for the specified number of seconds.
func (helper ScenarioHelper) Pause(secs int) {
    if !helper.isTestRun {
        time.Sleep(time.Duration(secs) * time.Second)
    }
}

// GetStackOutputs gets the outputs from the specified CloudFormation stack in a
structured format.
func (helper ScenarioHelper) GetStackOutputs(stackName string)
(actions.StackOutputs, error) {
    return helper.cfnActor.GetOutputs(stackName), nil
}

// PopulateUserTable fills the known user table with example data.
func (helper ScenarioHelper) PopulateUserTable(tableName string) {
    log.Printf("First, let's add some users to the DynamoDB %v table we'll use for
this example.\n", tableName)
    err := helper.dynamoActor.PopulateTable(tableName)
    if err != nil {
        panic(err)
    }
}

// GetKnownUsers gets the users from the known users table in a structured
format.
func (helper ScenarioHelper) GetKnownUsers(tableName string) (actions.UserList,
error) {
    knownUsers, err := helper.dynamoActor.Scan(tableName)
    if err != nil {
        log.Printf("Couldn't get known users from table %v. Here's why: %v\n",
tableName, err)
    }
    return knownUsers, err
}
```

```
}

// AddKnownUser adds a user to the known users table.
func (helper ScenarioHelper) AddKnownUser(tableName string, user actions.User) {
    log.Printf("Adding user '%v' with email '%v' to the DynamoDB known users
    table...\n",
        user.UserName, user.UserEmail)
    err := helper.dynamoActor.AddUser(tableName, user)
    if err != nil {
        panic(err)
    }
}

// ListRecentLogEvents gets the most recent log stream and events for the
// specified Lambda function and displays them.
func (helper ScenarioHelper) ListRecentLogEvents(functionName string) {
    log.Println("Waiting a few seconds to let Lambda write to CloudWatch Logs...")
    helper.Pause(10)
    log.Println("Okay, let's check the logs to find what's happened recently with
    your Lambda function.")
    logStream, err := helper.cwlActor.GetLatestLogStream(functionName)
    if err != nil {
        panic(err)
    }
    log.Printf("Getting some recent events from log stream %v\n",
        *logStream.LogStreamName)
    events, err := helper.cwlActor.GetLogEvents(functionName,
        *logStream.LogStreamName, 10)
    if err != nil {
        panic(err)
    }
    for _, event := range events {
        log.Printf("\t\t%v", *event.Message)
    }
    log.Println(strings.Repeat("-", 88))
}
```

Buat struct yang membungkus tindakan Amazon Cognito.

```
type CognitoActions struct {
    CognitoClient *cognitoidentityprovider.Client
}

// Trigger and TriggerInfo define typed data for updating an Amazon Cognito
trigger.
type Trigger int

const (
    PreSignUp Trigger = iota
    UserMigration
    PostAuthentication
)

type TriggerInfo struct {
    Trigger    Trigger
    HandlerArn *string
}

// UpdateTriggers adds or removes Lambda triggers for a user pool. When a trigger
is specified with a `nil` value,
// it is removed from the user pool.
func (actor CognitoActions) UpdateTriggers(userPoolId string,
    triggers ...TriggerInfo) error {
    output, err := actor.CognitoClient.DescribeUserPool(context.TODO(),
    &cognitoidentityprovider.DescribeUserPoolInput{
        UserPoolId: aws.String(userPoolId),
    })
    if err != nil {
        log.Printf("Couldn't get info about user pool %v. Here's why: %v\n",
        userPoolId, err)
        return err
    }
    lambdaConfig := output.UserPool.LambdaConfig
    for _, trigger := range triggers {
        switch trigger.Trigger {
        case PreSignUp:
            lambdaConfig.PreSignUp = trigger.HandlerArn
        case UserMigration:
            lambdaConfig.UserMigration = trigger.HandlerArn
        case PostAuthentication:
            lambdaConfig.PostAuthentication = trigger.HandlerArn
        }
    }
}
```

```
    }
  }
  _, err = actor.CognitoClient.UpdateUserPool(context.TODO(),
    &cognitoidentityprovider.UpdateUserPoolInput{
      UserPoolId:  aws.String(userPoolId),
      LambdaConfig: lambdaConfig,
    })
  if err != nil {
    log.Printf("Couldn't update user pool %v. Here's why: %v\n", userPoolId, err)
  }
  return err
}

// SignUp signs up a user with Amazon Cognito.
func (actor CognitoActions) SignUp(clientId string, userName string, password
string, userEmail string) (bool, error) {
  confirmed := false
  output, err := actor.CognitoClient.SignUp(context.TODO(),
    &cognitoidentityprovider.SignUpInput{
      ClientId: aws.String(clientId),
      Password: aws.String(password),
      Username: aws.String(userName),
      UserAttributes: []types.AttributeType{
        {Name: aws.String("email"), Value: aws.String(userEmail)},
      },
    })
  if err != nil {
    var invalidPassword *types.InvalidPasswordException
    if errors.As(err, &invalidPassword) {
      log.Println(*invalidPassword.Message)
    } else {
      log.Printf("Couldn't sign up user %v. Here's why: %v\n", userName, err)
    }
  } else {
    confirmed = output.UserConfirmed
  }
  return confirmed, err
}
```

```
// SignIn signs in a user to Amazon Cognito using a username and password
authentication flow.
func (actor CognitoActions) SignIn(clientId string, userName string, password
string) (*types.AuthenticationResultType, error) {
    var authResult *types.AuthenticationResultType
    output, err := actor.CognitoClient.InitiateAuth(context.TODO(),
    &cognitoidentityprovider.InitiateAuthInput{
        AuthFlow:      "USER_PASSWORD_AUTH",
        ClientId:      aws.String(clientId),
        AuthParameters: map[string]string{"USERNAME": userName, "PASSWORD": password},
    })
    if err != nil {
        var resetRequired *types.PasswordResetRequiredException
        if errors.As(err, &resetRequired) {
            log.Println(*resetRequired.Message)
        } else {
            log.Printf("Couldn't sign in user %v. Here's why: %v\n", userName, err)
        }
    } else {
        authResult = output.AuthenticationResult
    }
    return authResult, err
}

// ForgotPassword starts a password recovery flow for a user. This flow typically
sends a confirmation code
// to the user's configured notification destination, such as email.
func (actor CognitoActions) ForgotPassword(clientId string, userName string)
(*types.CodeDeliveryDetailsType, error) {
    output, err := actor.CognitoClient.ForgotPassword(context.TODO(),
    &cognitoidentityprovider.ForgotPasswordInput{
        ClientId: aws.String(clientId),
        Username: aws.String(userName),
    })
    if err != nil {
        log.Printf("Couldn't start password reset for user '%v'. Here;s why: %v\n",
        userName, err)
    }
    return output.CodeDeliveryDetails, err
}
```

```
// ConfirmForgotPassword confirms a user with a confirmation code and a new
password.
func (actor CognitoActions) ConfirmForgotPassword(clientId string, code string,
  userName string, password string) error {
  _, err := actor.CognitoClient.ConfirmForgotPassword(context.TODO(),
    &cognitoidentityprovider.ConfirmForgotPasswordInput{
      ClientId:      aws.String(clientId),
      ConfirmationCode: aws.String(code),
      Password:      aws.String(password),
      Username:      aws.String(userName),
    })
  if err != nil {
    var invalidPassword *types.InvalidPasswordException
    if errors.As(err, &invalidPassword) {
      log.Println(*invalidPassword.Message)
    } else {
      log.Printf("Couldn't confirm user %v. Here's why: %v", userName, err)
    }
  }
  return err
}

// DeleteUser removes a user from the user pool.
func (actor CognitoActions) DeleteUser(userAccessToken string) error {
  _, err := actor.CognitoClient.DeleteUser(context.TODO(),
    &cognitoidentityprovider.DeleteUserInput{
      AccessToken: aws.String(userAccessToken),
    })
  if err != nil {
    log.Printf("Couldn't delete user. Here's why: %v\n", err)
  }
  return err
}

// AdminCreateUser uses administrator credentials to add a user to a user pool.
This method leaves the user
// in a state that requires they enter a new password next time they sign in.
func (actor CognitoActions) AdminCreateUser(userPoolId string, userName string,
  userEmail string) error {
```

```
_, err := actor.CognitoClient.AdminCreateUser(context.TODO(),
&cognitoidentityprovider.AdminCreateUserInput{
    UserPoolId:    aws.String(userPoolId),
    Username:      aws.String(userName),
    MessageAction: types.MessageActionTypeSuppress,
    UserAttributes: []types.AttributeType{{Name: aws.String("email"), Value:
aws.String(userEmail)}}},
})
if err != nil {
    var userExists *types.UsernameExistsException
    if errors.As(err, &userExists) {
        log.Printf("User %v already exists in the user pool.", userName)
        err = nil
    } else {
        log.Printf("Couldn't create user %v. Here's why: %v\n", userName, err)
    }
}
return err
}

// AdminSetUserPassword uses administrator credentials to set a password for a
// user without requiring a
// temporary password.
func (actor CognitoActions) AdminSetUserPassword(userPoolId string, userName
string, password string) error {
    _, err := actor.CognitoClient.AdminSetUserPassword(context.TODO(),
&cognitoidentityprovider.AdminSetUserPasswordInput{
    Password:    aws.String(password),
    UserPoolId:  aws.String(userPoolId),
    Username:    aws.String(userName),
    Permanent:   true,
    })
    if err != nil {
        var invalidPassword *types.InvalidPasswordException
        if errors.As(err, &invalidPassword) {
            log.Println(*invalidPassword.Message)
        } else {
            log.Printf("Couldn't set password for user %v. Here's why: %v\n", userName,
err)
        }
    }
    return err
}
```



```
}
```

Buat struct yang membungkus tindakan DynamoDB.

```
// DynamoActions encapsulates the Amazon Simple Notification Service (Amazon SNS)
// actions
// used in the examples.
type DynamoActions struct {
    DynamoClient *dynamodb.Client
}

// User defines structured user data.
type User struct {
    UserName string
    UserEmail string
    LastLogin *LoginInfo `dynamodbav:",omitempty"`
}

// LoginInfo defines structured custom login data.
type LoginInfo struct {
    UserPoolId string
    ClientId string
    Time string
}

// UserList defines a list of users.
type UserList struct {
    Users []User
}

// UserNameList returns the usernames contained in a UserList as a list of
// strings.
func (users *UserList) UserNameList() []string {
    names := make([]string, len(users.Users))
    for i := 0; i < len(users.Users); i++ {
        names[i] = users.Users[i].UserName
    }
    return names
}
```

```
// PopulateTable adds a set of test users to the table.
func (actor DynamoActions) PopulateTable(tableName string) error {
    var err error
    var item map[string]types.AttributeValue
    var writeReqs []types.WriteRequest
    for i := 1; i < 4; i++ {
        item, err = attributevalue.MarshalMap(User{UserName: fmt.Sprintf("test_user_
%v", i), userEmail: fmt.Sprintf("test_email_%v@example.com", i)})
        if err != nil {
            log.Printf("Couldn't marshall user into DynamoDB format. Here's why: %v\n",
err)
            return err
        }
        writeReqs = append(writeReqs, types.WriteRequest{PutRequest:
&types.PutRequest{Item: item}})
    }
    _, err = actor.DynamoClient.BatchWriteItem(context.TODO(),
&dynamodb.BatchWriteItemInput{
    RequestItems: map[string][]types.WriteRequest{tableName: writeReqs},
})
    if err != nil {
        log.Printf("Couldn't populate table %v with users. Here's why: %v\n",
tableName, err)
    }
    return err
}

// Scan scans the table for all items.
func (actor DynamoActions) Scan(tableName string) (UserList, error) {
    var userList UserList
    output, err := actor.DynamoClient.Scan(context.TODO(), &dynamodb.ScanInput{
    TableName: aws.String(tableName),
})
    if err != nil {
        log.Printf("Couldn't scan table %v for items. Here's why: %v\n", tableName,
err)
    } else {
        err = attributevalue.UnmarshalListOfMaps(output.Items, &userList.Users)
        if err != nil {
            log.Printf("Couldn't unmarshal items into users. Here's why: %v\n", err)
        }
    }
    return userList, err
}
```

```
// AddUser adds a user item to a table.
func (actor DynamoActions) AddUser(tableName string, user User) error {
    userItem, err := attributevalue.MarshalMap(user)
    if err != nil {
        log.Printf("Couldn't marshall user to item. Here's why: %v\n", err)
    }
    _, err = actor.DynamoClient.PutItem(context.TODO(), &dynamodb.PutItemInput{
        Item:      userItem,
        TableName: aws.String(tableName),
    })
    if err != nil {
        log.Printf("Couldn't put item in table %v. Here's why: %v", tableName, err)
    }
    return err
}
```

Buat struct yang membungkus tindakan CloudWatch Log.

```
type CloudWatchLogsActions struct {
    CwlClient *cloudwatchlogs.Client
}

// GetLatestLogStream gets the most recent log stream for a Lambda function.
func (actor CloudWatchLogsActions) GetLatestLogStream(functionName string)
(types.LogStream, error) {
    var logStream types.LogStream
    logGroupName := fmt.Sprintf("/aws/lambda/%s", functionName)
    output, err := actor.CwlClient.DescribeLogStreams(context.TODO(),
&cloudwatchlogs.DescribeLogStreamsInput{
        Descending:  aws.Bool(true),
        Limit:        aws.Int32(1),
        LogGroupName: aws.String(logGroupName),
        OrderBy:     types.OrderByLastEventTime,
    })
    if err != nil {
        log.Printf("Couldn't get log streams for log group %v. Here's why: %v\n",
logGroupName, err)
    } else {
        logStream = output.LogStreams[0]
    }
}
```

```

    }
    return logStream, err
}

// GetLogEvents gets the most recent eventCount events from the specified log
// stream.
func (actor CloudWatchLogsActions) GetLogEvents(functionName string,
logStreamName string, eventCount int32) (
[]types.OutputLogEvent, error) {
var events []types.OutputLogEvent
logGroupName := fmt.Sprintf("/aws/lambda/%s", functionName)
output, err := actor.CwlClient.GetLogEvents(context.TODO(),
&cloudwatchlogs.GetLogEventsInput{
    LogStreamName: aws.String(logStreamName),
    Limit:         aws.Int32(eventCount),
    LogGroupName:  aws.String(logGroupName),
})
if err != nil {
    log.Printf("Couldn't get log event for log stream %v. Here's why: %v\n",
logStreamName, err)
} else {
    events = output.Events
}
return events, err
}

```

Buat struct yang membungkus tindakan AWS CloudFormation .

```

// StackOutputs defines a map of outputs from a specific stack.
type StackOutputs map[string]string

type CloudFormationActions struct {
    CfnClient *cloudformation.Client
}

// GetOutputs gets the outputs from a CloudFormation stack and puts them into a
// structured format.
func (actor CloudFormationActions) GetOutputs(stackName string) StackOutputs {
    output, err := actor.CfnClient.DescribeStacks(context.TODO(),
    &cloudformation.DescribeStacksInput{

```

```

    StackName: aws.String(stackName),
  })
  if err != nil || len(output.Stacks) == 0 {
    log.Panicf("Couldn't find a CloudFormation stack named %v. Here's why: %v\n",
      stackName, err)
  }
  stackOutputs := StackOutputs{}
  for _, out := range output.Stacks[0].Outputs {
    stackOutputs[*out.OutputKey] = *out.OutputValue
  }
  return stackOutputs
}

```

## Pembersihan sumber daya

```

// Resources keeps track of AWS resources created during an example and handles
// cleanup when the example finishes.
type Resources struct {
  userPoolId      string
  userAccessTokens []string
  triggers        []actions.Trigger

  cognitoActor *actions.CognitoActions
  questioner   demotools.IQuestioner
}

func (resources *Resources) init(cognitoActor *actions.CognitoActions, questioner
  demotools.IQuestioner) {
  resources.userAccessTokens = []string{}
  resources.triggers = []actions.Trigger{}
  resources.cognitoActor = cognitoActor
  resources.questioner = questioner
}

// Cleanup deletes all AWS resources created during an example.
func (resources *Resources) Cleanup() {
  defer func() {
    if r := recover(); r != nil {
      log.Printf("Something went wrong during cleanup.\n%v\n", r)
    }
  }()
}

```

```
    log.Println("Use the AWS Management Console to remove any remaining resources\n" +\n        "that were created for this scenario.")\n    }\n}\n\nwantDelete := resources.questioner.AskBool("Do you want to remove all of the AWS\nresources that were created "+\n    "during this demo (y/n)?", "y")\nif wantDelete {\n    for _, accessToken := range resources.userAccessTokens {\n        err := resources.cognitoActor.DeleteUser(accessToken)\n        if err != nil {\n            log.Println("Couldn't delete user during cleanup.")\n            panic(err)\n        }\n        log.Println("Deleted user.")\n    }\n    triggerList := make([]actions.TriggerInfo, len(resources.triggers))\n    for i := 0; i < len(resources.triggers); i++ {\n        triggerList[i] = actions.TriggerInfo{Trigger: resources.triggers[i],\nHandlerArn: nil}\n    }\n    err := resources.cognitoActor.UpdateTriggers(resources.userPoolId,\ntriggerList...)\n    if err != nil {\n        log.Println("Couldn't update Cognito triggers during cleanup.")\n        panic(err)\n    }\n    log.Println("Removed Cognito triggers from user pool.")\n} else {\n    log.Println("Be sure to remove resources when you're done with them to avoid\nunexpected charges!")\n}\n}
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK for Go .
  - [ConfirmForgotPassword](#)
  - [DeleteUser](#)
  - [ForgotPassword](#)

- [InitiateAuth](#)
- [SignUp](#)
- [UpdateUserPool](#)

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Mendaftar pengguna dengan kumpulan pengguna Amazon Cognito yang memerlukan MFA menggunakan SDK AWS

Contoh kode berikut ini menunjukkan cara:

- Daftar dan konfirmasi pengguna dengan nama pengguna, kata sandi, dan alamat email.
- Siapkan otentikasi multi-faktor dengan mengaitkan aplikasi MFA dengan pengguna.
- Masuk dengan menggunakan kata sandi dan kode MFA.

### .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 CognitoBasics;

public class CognitoBasics
{
    private static ILogger logger = null!;

    static async Task Main(string[] args)
    {
        // Set up dependency injection for Amazon Cognito.
        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<IAmazonCognitoIdentityProvider>()
.AddTransient<CognitoWrapper>()
)
        .Build();

logger = LoggerFactory.Create(builder => { builder.AddConsole(); })
    .CreateLogger<CognitoBasics>();

var configuration = new ConfigurationBuilder()
    .SetBasePath(Directory.GetCurrentDirectory())
    .AddJsonFile("settings.json") // Load settings from .json file.
    .AddJsonFile("settings.local.json",
        true) // Optionally load local settings.
    .Build();

var cognitoWrapper = host.Services.GetRequiredService<CognitoWrapper>();

Console.WriteLine(new string('-', 80));
UiMethods.DisplayOverview();
Console.WriteLine(new string('-', 80));

// clientId - The app client Id value that you get from the AWS CDK
script.
var clientId = configuration["ClientId"]; // **** REPLACE WITH CLIENT ID
VALUE FROM CDK SCRIPT";

// poolId - The pool Id that you get from the AWS CDK script.
var poolId = configuration["PoolId"]!; // **** REPLACE WITH POOL ID VALUE
FROM CDK SCRIPT";
var userName = configuration["UserName"];
var password = configuration["Password"];
var email = configuration["Email"];

// If the username wasn't set in the configuration file,
// get it from the user now.
if (userName is null)
{
    do
```



```
        {
            Console.WriteLine("Username: ");
            userName = Console.ReadLine();
        }
        while (string.IsNullOrEmpty(userName));
    }
    Console.WriteLine($"\\nUsername: {userName}");

    // If the password wasn't set in the configuration file,
    // get it from the user now.
    if (password is null)
    {
        do
        {
            Console.WriteLine("Password: ");
            password = Console.ReadLine();
        }
        while (string.IsNullOrEmpty(password));
    }

    // If the email address wasn't set in the configuration file,
    // get it from the user now.
    if (email is null)
    {
        do
        {
            Console.WriteLine("Email: ");
            email = Console.ReadLine();
        } while (string.IsNullOrEmpty(email));
    }

    // Now sign up the user.
    Console.WriteLine($"\\nSigning up {userName} with email address:
{email}");
    await cognitoWrapper.SignUpAsync(clientId, userName, password, email);

    // Add the user to the user pool.
    Console.WriteLine($"Adding {userName} to the user pool");
    await cognitoWrapper.GetAdminUserAsync(userName, poolId);

    UiMethods.DisplayTitle("Get confirmation code");
    Console.WriteLine($"Conformation code sent to {userName}.");
    Console.WriteLine("Would you like to send a new code? (Y/N) ");
    var answer = Console.ReadLine();
```

```
if (answer!.ToLower() == "y")
{
    await cognitoWrapper.ResendConfirmationCodeAsync(clientId, userName);
    Console.WriteLine("Sending a new confirmation code");
}

Console.Write("Enter confirmation code (from Email): ");
var code = Console.ReadLine();

await cognitoWrapper.ConfirmSignupAsync(clientId, code, userName);

UiMethods.DisplayTitle("Checking status");
Console.WriteLine($"Rechecking the status of {userName} in the user
pool");
await cognitoWrapper.GetAdminUserAsync(userName, poolId);

Console.WriteLine($"Setting up authenticator for {userName} in the user
pool");
var setupResponse = await cognitoWrapper.InitiateAuthAsync(clientId,
userName, password);

var setupSession = await
cognitoWrapper.AssociateSoftwareTokenAsync(setupResponse.Session);
Console.Write("Enter the 6-digit code displayed in Google Authenticator:
");
var setupCode = Console.ReadLine();

var setupResult = await
cognitoWrapper.VerifySoftwareTokenAsync(setupSession, setupCode);
Console.WriteLine($"Setup status: {setupResult}");

Console.WriteLine($"Now logging in {userName} in the user pool");
var authSession = await cognitoWrapper.AdminInitiateAuthAsync(clientId,
poolId, userName, password);

Console.Write("Enter a new 6-digit code displayed in Google
Authenticator: ");
var authCode = Console.ReadLine();

var authResult = await
cognitoWrapper.AdminRespondToAuthChallengeAsync(userName, clientId, authCode,
authSession, poolId);
```

```
        Console.WriteLine($"Authenticated and received access token:
{authResult.AccessToken}");

        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Cognito scenario is complete.");
        Console.WriteLine(new string('-', 80));
    }
}

using System.Net;

namespace CognitoActions;

/// <summary>
/// Methods to perform Amazon Cognito Identity Provider actions.
/// </summary>
public class CognitoWrapper
{
    private readonly IAmazonCognitoIdentityProvider _cognitoService;

    /// <summary>
    /// Constructor for the wrapper class containing Amazon Cognito actions.
    /// </summary>
    /// <param name="cognitoService">The Amazon Cognito client object.</param>
    public CognitoWrapper(IAmazonCognitoIdentityProvider cognitoService)
    {
        _cognitoService = cognitoService;
    }

    /// <summary>
    /// List the Amazon Cognito user pools for an account.
    /// </summary>
    /// <returns>A list of UserPoolDescriptionType objects.</returns>
    public async Task<List<UserPoolDescriptionType>> ListUserPoolsAsync()
    {
        var userPools = new List<UserPoolDescriptionType>();

        var userPoolsPaginator = _cognitoService.Paginators.ListUserPools(new
ListUserPoolsRequest());

        await foreach (var response in userPoolsPaginator.Responses)
        {
            userPools.AddRange(response.UserPools);
        }
    }
}
```

```
    }

    return userPools;
}

/// <summary>
/// Get a list of users for the Amazon Cognito user pool.
/// </summary>
/// <param name="userPoolId">The user pool ID.</param>
/// <returns>A list of users.</returns>
public async Task<List<UserType>> ListUsersAsync(string userPoolId)
{
    var request = new ListUsersRequest
    {
        UserPoolId = userPoolId
    };

    var users = new List<UserType>();

    var usersPaginator = _cognitoService.Paginators.ListUsers(request);
    await foreach (var response in usersPaginator.Responses)
    {
        users.AddRange(response.Users);
    }

    return users;
}

/// <summary>
/// Respond to an admin authentication challenge.
/// </summary>
/// <param name="userName">The name of the user.</param>
/// <param name="clientId">The client ID.</param>
/// <param name="mfaCode">The multi-factor authentication code.</param>
/// <param name="session">The current application session.</param>
/// <param name="clientId">The user pool ID.</param>
/// <returns>The result of the authentication response.</returns>
public async Task<AuthenticationResultType> AdminRespondToAuthChallengeAsync(
    string userName,
    string clientId,
    string mfaCode,
    string session,
```

```
        string userPoolId)
    {
        Console.WriteLine("SOFTWARE_TOKEN_MFA challenge is generated");

        var challengeResponses = new Dictionary<string, string>();
        challengeResponses.Add("USERNAME", userName);
        challengeResponses.Add("SOFTWARE_TOKEN_MFA_CODE", mfaCode);

        var respondToAuthChallengeRequest = new
AdminRespondToAuthChallengeRequest
        {
            ChallengeName = ChallengeNameType.SOFTWARE_TOKEN_MFA,
            ClientId = clientId,
            ChallengeResponses = challengeResponses,
            Session = session,
            UserPoolId = userPoolId,
        };

        var response = await
_cognitoService.AdminRespondToAuthChallengeAsync(respondToAuthChallengeRequest);
        Console.WriteLine($"Response to Authentication
{response.AuthenticationResult.TokenType}");
        return response.AuthenticationResult;
    }

    /// <summary>
    /// Verify the TOTP and register for MFA.
    /// </summary>
    /// <param name="session">The name of the session.</param>
    /// <param name="code">The MFA code.</param>
    /// <returns>The status of the software token.</returns>
    public async Task<VerifySoftwareTokenResponseType>
VerifySoftwareTokenAsync(string session, string code)
    {
        var tokenRequest = new VerifySoftwareTokenRequest
        {
            UserCode = code,
            Session = session,
        };

        var verifyResponse = await
_cognitoService.VerifySoftwareTokenAsync(tokenRequest);
```

```
        return verifyResponse.Status;
    }

    /// <summary>
    /// Get an MFA token to authenticate the user with the authenticator.
    /// </summary>
    /// <param name="session">The session name.</param>
    /// <returns>The session name.</returns>
    public async Task<string> AssociateSoftwareTokenAsync(string session)
    {
        var softwareTokenRequest = new AssociateSoftwareTokenRequest
        {
            Session = session,
        };

        var tokenResponse = await
        _cognitoService.AssociateSoftwareTokenAsync(softwareTokenRequest);
        var secretCode = tokenResponse.SecretCode;

        Console.WriteLine($"Use the following secret code to set up the
        authenticator: {secretCode}");

        return tokenResponse.Session;
    }

    /// <summary>
    /// Initiate an admin auth request.
    /// </summary>
    /// <param name="clientId">The client ID to use.</param>
    /// <param name="userPoolId">The ID of the user pool.</param>
    /// <param name="userName">The username to authenticate.</param>
    /// <param name="password">The user's password.</param>
    /// <returns>The session to use in challenge-response.</returns>
    public async Task<string> AdminInitiateAuthAsync(string clientId, string
    userPoolId, string userName, string password)
    {
        var authParameters = new Dictionary<string, string>();
        authParameters.Add("USERNAME", userName);
        authParameters.Add("PASSWORD", password);

        var request = new AdminInitiateAuthRequest
        {
```

```
        ClientId = clientId,
        UserPoolId = userPoolId,
        AuthParameters = authParameters,
        AuthFlow = AuthFlowType.ADMIN_USER_PASSWORD_AUTH,
    };

    var response = await _cognitoService.AdminInitiateAuthAsync(request);
    return response.Session;
}

/// <summary>
/// Initiate authorization.
/// </summary>
/// <param name="clientId">The client Id of the application.</param>
/// <param name="userName">The name of the user who is authenticating.</
param>
/// <param name="password">The password for the user who is authenticating.</
param>
/// <returns>The response from the initiate auth request.</returns>
public async Task<InitiateAuthResponse> InitiateAuthAsync(string clientId,
string userName, string password)
{
    var authParameters = new Dictionary<string, string>();
    authParameters.Add("USERNAME", userName);
    authParameters.Add("PASSWORD", password);

    var authRequest = new InitiateAuthRequest

    {
        ClientId = clientId,
        AuthParameters = authParameters,
        AuthFlow = AuthFlowType.USER_PASSWORD_AUTH,
    };

    var response = await _cognitoService.InitiateAuthAsync(authRequest);
    Console.WriteLine($"Result Challenge is : {response.ChallengeName}");

    return response;
}

/// <summary>
/// Confirm that the user has signed up.
/// </summary>
/// <param name="clientId">The Id of this application.</param>
```

```
/// <param name="code">The confirmation code sent to the user.</param>
/// <param name="userName">The username.</param>
/// <returns>True if successful.</returns>
public async Task<bool> ConfirmSignUpAsync(string clientId, string code,
string userName)
{
    var signUpRequest = new ConfirmSignUpRequest
    {
        ClientId = clientId,
        ConfirmationCode = code,
        Username = userName,
    };

    var response = await _cognitoService.ConfirmSignUpAsync(signUpRequest);
    if (response.HttpStatusCode == HttpStatusCode.OK)
    {
        Console.WriteLine($"{userName} was confirmed");
        return true;
    }
    return false;
}

/// <summary>
/// Initiates and confirms tracking of the device.
/// </summary>
/// <param name="accessToken">The user's access token.</param>
/// <param name="deviceKey">The key of the device from Amazon Cognito.</
param>
/// <param name="deviceName">The device name.</param>
/// <returns></returns>
public async Task<bool> ConfirmDeviceAsync(string accessToken, string
deviceKey, string deviceName)
{
    var request = new ConfirmDeviceRequest
    {
        AccessToken = accessToken,
        DeviceKey = deviceKey,
        DeviceName = deviceName
    };

    var response = await _cognitoService.ConfirmDeviceAsync(request);
    return response.UserConfirmationNecessary;
}
```



```
    /// <summary>
    /// Send a new confirmation code to a user.
    /// </summary>
    /// <param name="clientId">The Id of the client application.</param>
    /// <param name="userName">The username of user who will receive the code.</
param>
    /// <returns>The delivery details.</returns>
    public async Task<CodeDeliveryDetailsType> ResendConfirmationCodeAsync(string
clientId, string userName)
    {
        var codeRequest = new ResendConfirmationCodeRequest
        {
            ClientId = clientId,
            Username = userName,
        };

        var response = await
_cognitoService.ResendConfirmationCodeAsync(codeRequest);

        Console.WriteLine($"Method of delivery is
{response.CodeDeliveryDetails.DeliveryMedium}");

        return response.CodeDeliveryDetails;
    }

    /// <summary>
    /// Get the specified user from an Amazon Cognito user pool with
administrator access.
    /// </summary>
    /// <param name="userName">The name of the user.</param>
    /// <param name="poolId">The Id of the Amazon Cognito user pool.</param>
    /// <returns>Async task.</returns>
    public async Task<UserStatusType> GetAdminUserAsync(string userName, string
poolId)
    {
        AdminGetUserRequest userRequest = new AdminGetUserRequest
        {
            Username = userName,
            UserPoolId = poolId,
        };
    }
}
```

```
        var response = await _cognitoService.AdminGetUserAsync(userRequest);

        Console.WriteLine($"User status {response.UserStatus}");
        return response.UserStatus;
    }

    /// <summary>
    /// Sign up a new user.
    /// </summary>
    /// <param name="clientId">The client Id of the application.</param>
    /// <param name="userName">The username to use.</param>
    /// <param name="password">The user's password.</param>
    /// <param name="email">The email address of the user.</param>
    /// <returns>A Boolean value indicating whether the user was confirmed.</
returns>
    public async Task<bool> SignUpAsync(string clientId, string userName, string
password, string email)
    {
        var userAttrs = new AttributeType
        {
            Name = "email",
            Value = email,
        };

        var userAttrsList = new List<AttributeType>();

        userAttrsList.Add(userAttrs);

        var signUpRequest = new SignUpRequest
        {
            UserAttributes = userAttrsList,
            Username = userName,
            ClientId = clientId,
            Password = password
        };

        var response = await _cognitoService.SignUpAsync(signUpRequest);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
}
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK for .NET .
  - [AdminGetUser](#)
  - [AdminInitiateAuth](#)
  - [AdminRespondToAuthChallenge](#)
  - [AssociateSoftwareToken](#)
  - [ConfirmDevice](#)
  - [ConfirmSignUp](#)
  - [InitiateAuth](#)
  - [ListUsers](#)
  - [ResendConfirmationCode](#)
  - [RespondToAuthChallenge](#)
  - [SignUp](#)
  - [VerifySoftwareToken](#)

## C++

### SDK untuk C++

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
Aws::Client::ClientConfiguration clientConfig;
// Optional: Set to the AWS Region (overrides config file).
// clientConfig.region = "us-east-1";

//! Scenario that adds a user to an Amazon Cognito user pool.
/*!
\sa gettingStartedWithUserPools()
\param clientID: Client ID associated with an Amazon Cognito user pool.
\param userPoolID: An Amazon Cognito user pool ID.
\param clientConfig: Aws client configuration.
```

```
\return bool: Successful completion.
*/
bool AwsDoc::Cognito::gettingStartedWithUserPools(const Aws::String &clientID,
                                                    const Aws::String &userPoolID,
                                                    const
                                                    Aws::Client::ClientConfiguration &clientConfig) {
    printAsterisksLine();
    std::cout
        << "Welcome to the Amazon Cognito example scenario."
        << std::endl;
    printAsterisksLine();

    std::cout
        << "This scenario will add a user to an Amazon Cognito user pool."
        << std::endl;
    const Aws::String userName = askQuestion("Enter a new username: ");
    const Aws::String password = askQuestion("Enter a new password: ");
    const Aws::String email = askQuestion("Enter a valid email for the user: ");

    std::cout << "Signing up " << userName << std::endl;

    Aws::CognitoIdentityProvider::CognitoIdentityProviderClient
client(clientConfig);
    bool userExists = false;
    do {
        // 1. Add a user with a username, password, and email address.
        Aws::CognitoIdentityProvider::Model::SignUpRequest request;
        request.AddUserAttributes(
            Aws::CognitoIdentityProvider::Model::AttributeType().WithName(
                "email").WithValue(email));
        request.SetUsername(userName);
        request.SetPassword(password);
        request.SetClientId(clientID);
        Aws::CognitoIdentityProvider::Model::SignUpOutcome outcome =
            client.SignUp(request);

        if (outcome.IsSuccess()) {
            std::cout << "The signup request for " << userName << " was
successful."
                << std::endl;
        }
        else if (outcome.GetError().GetErrorType() ==
Aws::CognitoIdentityProvider::CognitoIdentityProviderErrors::USERNAME_EXISTS) {
```

```
        std::cout
            << "The username already exists. Please enter a different
username."
            << std::endl;
        userExists = true;
    }
    else {
        std::cerr << "Error with CognitoIdentityProvider::SignUpRequest. "
            << outcome.GetError().GetMessage()
            << std::endl;
        return false;
    }
} while (userExists);

printAsterisksLine();
std::cout << "Retrieving status of " << userName << " in the user pool."
    << std::endl;
// 2. Confirm that the user was added to the user pool.
if (!checkAdminUserStatus(userName, userPoolID, client)) {
    return false;
}

std::cout << "A confirmation code was sent to " << email << "." << std::endl;

bool resend = askYesNoQuestion("Would you like to send a new code? (y/n) ");
if (resend) {
    // Request a resend of the confirmation code to the email address.
    (ResendConfirmationCode)
    Aws::CognitoIdentityProvider::Model::ResendConfirmationCodeRequest
request;
    request.SetUsername(userName);
    request.SetClientId(clientID);

    Aws::CognitoIdentityProvider::Model::ResendConfirmationCodeOutcome
outcome =
        client.ResendConfirmationCode(request);

    if (outcome.IsSuccess()) {
        std::cout
            << "CognitoIdentityProvider::ResendConfirmationCode was
successful."
            << std::endl;
    }
    else {
```

```
        std::cerr << "Error with
CognitoIdentityProvider::ResendConfirmationCode. "
                << outcome.GetError().GetMessage()
                << std::endl;
        return false;
    }
}

printAsterisksLine();

{
    // 4. Send the confirmation code that's received in the email.
    (ConfirmSignUp)
    const Aws::String confirmationCode = askQuestion(
        "Enter the confirmation code that was emailed: ");
    Aws::CognitoIdentityProvider::Model::ConfirmSignUpRequest request;
    request.SetClientId(clientID);
    request.SetConfirmationCode(confirmationCode);
    request.SetUsername(userName);

    Aws::CognitoIdentityProvider::Model::ConfirmSignUpOutcome outcome =
        client.ConfirmSignUp(request);

    if (outcome.IsSuccess()) {
        std::cout << "ConfirmSignup was Successful."
                << std::endl;
    }
    else {
        std::cerr << "Error with CognitoIdentityProvider::ConfirmSignUp. "
                << outcome.GetError().GetMessage()
                << std::endl;
        return false;
    }
}

std::cout << "Rechecking the status of " << userName << " in the user pool."
        << std::endl;
if (!checkAdminUserStatus(userName, userPoolID, client)) {
    return false;
}

printAsterisksLine();

std::cout << "Initiating authorization using the username and password."
```

```

        << std::endl;

    Aws::String session;
    // 5. Initiate authorization with username and password. (AdminInitiateAuth)
    if (!adminInitiateAuthorization(clientID, userPoolID, userName, password,
    session, client)) {
        return false;
    }

    printAsterisksLine();

    std::cout
        << "Starting setup of time-based one-time password (TOTP) multi-
factor authentication (MFA)."
        << std::endl;

    {
        // 6. Request a setup key for one-time password (TOTP)
        // multi-factor authentication (MFA). (AssociateSoftwareToken)
        Aws::CognitoIdentityProvider::Model::AssociateSoftwareTokenRequest
request;
        request.SetSession(session);

        Aws::CognitoIdentityProvider::Model::AssociateSoftwareTokenOutcome
outcome =
            client.AssociateSoftwareToken(request);

        if (outcome.IsSuccess()) {
            std::cout
                << "Enter this setup key into an authenticator app, for
example Google Authenticator."
                << std::endl;
            std::cout << "Setup key: " << outcome.GetResult().GetSecretCode()
                << std::endl;
#ifdef USING_QR
            printAsterisksLine();
            std::cout << "\nOr scan the QR code in the file '" << QR_CODE_PATH <<
            "."
                << std::endl;

            saveQRCode(std::string("otpauth://totp/") + userName + "?secret=" +
                outcome.GetResult().GetSecretCode());
#endif // USING_QR
            session = outcome.GetResult().GetSession();

```

```
    }
    else {
        std::cerr << "Error with
CognitoIdentityProvider::AssociateSoftwareToken. "
                << outcome.GetError().GetMessage()
                << std::endl;
        return false;
    }
}
askQuestion("Type enter to continue...", alwaysTrueTest);

printAsterisksLine();

{
    Aws::String userCode = askQuestion(
        "Enter the 6 digit code displayed in the authenticator app: ");

    // 7. Send the MFA code copied from an authenticator app.
(VerifySoftwareToken)
    Aws::CognitoIdentityProvider::Model::VerifySoftwareTokenRequest request;
    request.SetUserCode(userCode);
    request.SetSession(session);

    Aws::CognitoIdentityProvider::Model::VerifySoftwareTokenOutcome outcome =
        client.VerifySoftwareToken(request);

    if (outcome.IsSuccess()) {
        std::cout << "Verification of the code was successful."
                << std::endl;
        session = outcome.GetResult().GetSession();
    }
    else {
        std::cerr << "Error with
CognitoIdentityProvider::VerifySoftwareToken. "
                << outcome.GetError().GetMessage()
                << std::endl;
        return false;
    }
}

printAsterisksLine();
std::cout << "You have completed the MFA authentication setup." << std::endl;
std::cout << "Now, sign in." << std::endl;
```



```
// 8. Initiate authorization again with username and password.
(AdminInitiateAuth)
    if (!adminInitiateAuthorization(clientID, userPoolID, userName, password,
session, client)) {
        return false;
    }

    Aws::String accessToken;
    {
        Aws::String mfaCode = askQuestion(
            "Re-enter the 6 digit code displayed in the authenticator app:
");

        // 9. Send a new MFA code copied from an authenticator app.
(AdminRespondToAuthChallenge)
        Aws::CognitoIdentityProvider::Model::AdminRespondToAuthChallengeRequest
request;
        request.AddChallengeResponses("USERNAME", userName);
        request.AddChallengeResponses("SOFTWARE_TOKEN_MFA_CODE", mfaCode);
        request.SetChallengeName(

Aws::CognitoIdentityProvider::Model::ChallengeNameType::SOFTWARE_TOKEN_MFA);
        request.SetClientId(clientID);
        request.SetUserPoolId(userPoolID);
        request.SetSession(session);

        Aws::CognitoIdentityProvider::Model::AdminRespondToAuthChallengeOutcome
outcome =
            client.AdminRespondToAuthChallenge(request);

        if (outcome.IsSuccess()) {
            std::cout << "Here is the response to the challenge.\n" <<

outcome.GetResult().GetAuthenticationResult().Jsonize().View().WriteReadable()
            << std::endl;

            accessToken =
outcome.GetResult().GetAuthenticationResult().GetAccessToken();
        }
        else {
            std::cerr << "Error with
CognitoIdentityProvider::AdminRespondToAuthChallenge. "
                << outcome.GetError().GetMessage()
                << std::endl;
        }
    }
}
```

```

        return false;
    }

    std::cout << "You have successfully added a user to Amazon Cognito."
              << std::endl;
}

if (askYesNoQuestion("Would you like to delete the user that you just added?
(y/n) ")) {
    // 10. Delete the user that you just added. (DeleteUser)
    Aws::CognitoIdentityProvider::Model::DeleteUserRequest request;
    request.SetAccessToken(accessToken);

    Aws::CognitoIdentityProvider::Model::DeleteUserOutcome outcome =
        client.DeleteUser(request);

    if (outcome.IsSuccess()) {
        std::cout << "The user " << userName << " was deleted."
                  << std::endl;
    }
    else {
        std::cerr << "Error with CognitoIdentityProvider::DeleteUser. "
                  << outcome.GetError().GetMessage()
                  << std::endl;
    }
}

return true;
}

//! Routine which checks the user status in an Amazon Cognito user pool.
/*!
 \sa checkAdminUserStatus()
 \param userName: A username.
 \param userPoolID: An Amazon Cognito user pool ID.
 \return bool: Successful completion.
 */
bool AwsDoc::Cognito::checkAdminUserStatus(const Aws::String &userName,
                                           const Aws::String &userPoolID,
                                           const
    Aws::CognitoIdentityProvider::CognitoIdentityProviderClient &client) {
    Aws::CognitoIdentityProvider::Model::AdminGetUserRequest request;
    request.SetUsername(userName);
    request.SetUserPoolId(userPoolID);

```

```

    Aws::CognitoIdentityProvider::Model::AdminGetUserOutcome outcome =
        client.AdminGetUser(request);

    if (outcome.IsSuccess()) {
        std::cout << "The status for " << userName << " is " <<

Aws::CognitoIdentityProvider::Model::UserStatusTypeMapper::GetNameForUserStatusType(
        outcome.GetResult().GetUserStatus()) << std::endl;
        std::cout << "Enabled is " << outcome.GetResult().GetEnabled() <<
std::endl;
    }
    else {
        std::cerr << "Error with CognitoIdentityProvider::AdminGetUser. "
            << outcome.GetError().GetMessage()
            << std::endl;
    }

    return outcome.IsSuccess();
}

//! Routine which starts authorization of an Amazon Cognito user.
//! This routine requires administrator credentials.
/*!
 \sa adminInitiateAuthorization()
 \param clientID: Client ID of tracked device.
 \param userPoolID: An Amazon Cognito user pool ID.
 \param userName: A username.
 \param password: A password.
 \param sessionResult: String to receive a session token.
 \return bool: Successful completion.
 */
bool AwsDoc::Cognito::adminInitiateAuthorization(const Aws::String &clientID,
                                                const Aws::String &userPoolID,
                                                const Aws::String &userName,
                                                const Aws::String &password,
                                                Aws::String &sessionResult,
                                                const
    Aws::CognitoIdentityProvider::CognitoIdentityProviderClient &client) {
    Aws::CognitoIdentityProvider::Model::AdminInitiateAuthRequest request;
    request.SetClientId(clientID);
    request.SetUserPoolId(userPoolID);
    request.AddAuthParameters("USERNAME", userName);
    request.AddAuthParameters("PASSWORD", password);

```

```
request.SetAuthFlow(
    Aws::CognitoIdentityProvider::Model::AuthFlowType::ADMIN_USER_PASSWORD_AUTH);

    Aws::CognitoIdentityProvider::Model::AdminInitiateAuthOutcome outcome =
        client.AdminInitiateAuth(request);

    if (outcome.IsSuccess()) {
        std::cout << "Call to AdminInitiateAuth was successful." << std::endl;
        sessionResult = outcome.GetResult().GetSession();
    }
    else {
        std::cerr << "Error with CognitoIdentityProvider::AdminInitiateAuth. "
            << outcome.GetError().GetMessage()
            << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK for C++ .
  - [AdminGetUser](#)
  - [AdminInitiateAuth](#)
  - [AdminRespondToAuthChallenge](#)
  - [AssociateSoftwareToken](#)
  - [ConfirmDevice](#)
  - [ConfirmSignUp](#)
  - [InitiateAuth](#)
  - [ListUsers](#)
  - [ResendConfirmationCode](#)
  - [RespondToAuthChallenge](#)
  - [SignUp](#)
  - [VerifySoftwareToken](#)

## Java

### SDK untuk Java 2.x

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
import software.amazon.awssdk.regions.Region;
import
    software.amazon.awssdk.services.cognitoidentityprovider.CognitoIdentityProviderClient;
import
    software.amazon.awssdk.services.cognitoidentityprovider.model.AdminGetUserRequest;
import
    software.amazon.awssdk.services.cognitoidentityprovider.model.AdminGetUserResponse;
import
    software.amazon.awssdk.services.cognitoidentityprovider.model.AdminInitiateAuthRequest;
import
    software.amazon.awssdk.services.cognitoidentityprovider.model.AdminInitiateAuthResponse;
import
    software.amazon.awssdk.services.cognitoidentityprovider.model.AdminRespondToAuthChallenge;
import
    software.amazon.awssdk.services.cognitoidentityprovider.model.AdminRespondToAuthChallengeResponse;
import
    software.amazon.awssdk.services.cognitoidentityprovider.model.AssociateSoftwareTokenRequest;
import
    software.amazon.awssdk.services.cognitoidentityprovider.model.AssociateSoftwareTokenResponse;
import
    software.amazon.awssdk.services.cognitoidentityprovider.model.AttributeType;
import
    software.amazon.awssdk.services.cognitoidentityprovider.model.AuthFlowType;
import
    software.amazon.awssdk.services.cognitoidentityprovider.model.ChallengeNameType;
import
    software.amazon.awssdk.services.cognitoidentityprovider.model.CognitoIdentityProviderException;
import
    software.amazon.awssdk.services.cognitoidentityprovider.model.ConfirmSignUpRequest;
import
    software.amazon.awssdk.services.cognitoidentityprovider.model.ResendConfirmationCodeRequest;
```

```
import
    software.amazon.awssdk.services.cognitoidentityprovider.model.ResendConfirmationCodeResp
import
    software.amazon.awssdk.services.cognitoidentityprovider.model.SignUpRequest;
import
    software.amazon.awssdk.services.cognitoidentityprovider.model.VerifySoftwareTokenRequest
import
    software.amazon.awssdk.services.cognitoidentityprovider.model.VerifySoftwareTokenResponse
import java.security.InvalidKeyException;
import java.security.NoSuchAlgorithmException;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import java.util.Scanner;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 *
 * TIP: To set up the required user pool, run the AWS Cloud Development Kit (AWS
 * CDK) script provided in this GitHub repo at
 * resources/cdk/cognito\_scenario\_user\_pool\_with\_mfa.
 *
 * This code example performs the following operations:
 *
 * 1. Invokes the signUp method to sign up a user.
 * 2. Invokes the adminGetUser method to get the user's confirmation status.
 * 3. Invokes the ResendConfirmationCode method if the user requested another
 * code.
 * 4. Invokes the confirmSignUp method.
 * 5. Invokes the AdminInitiateAuth to sign in. This results in being prompted
 * to set up TOTP (time-based one-time password). (The response is
 * "ChallengeName": "MFA_SETUP").
 * 6. Invokes the AssociateSoftwareToken method to generate a TOTP MFA private
 * key. This can be used with Google Authenticator.
 * 7. Invokes the VerifySoftwareToken method to verify the TOTP and register for
 * MFA.
 * 8. Invokes the AdminInitiateAuth to sign in again. This results in being
```

```
* prompted to submit a TOTP (Response: "ChallengeName": "SOFTWARE_TOKEN_MFA").
* 9. Invokes the AdminRespondToAuthChallenge to get back a token.
*/

public class CognitoMVP {
    public static final String DASHES = new String(new char[80]).replace("\0",
    "-");

    public static void main(String[] args) throws NoSuchAlgorithmException,
    InvalidKeyException {
        final String usage = ""

            Usage:
                <clientId> <poolId>

            Where:
                clientId - The app client Id value that you can get from the
AWS CDK script.
                poolId - The pool Id that you can get from the AWS CDK
script.\s
            """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String clientId = args[0];
        String poolId = args[1];
        CognitoIdentityProviderClient identityProviderClient =
CognitoIdentityProviderClient.builder()
            .region(Region.US_EAST_1)
            .build();

        System.out.println(DASHES);
        System.out.println("Welcome to the Amazon Cognito example scenario.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("*** Enter your user name");
        Scanner in = new Scanner(System.in);
        String userName = in.nextLine();

        System.out.println("*** Enter your password");
```

```
String password = in.nextLine();

System.out.println("*** Enter your email");
String email = in.nextLine();

System.out.println("1. Signing up " + userName);
signUp(identityProviderClient, clientId, userName, password, email);
System.out.println(DASHES);

System.out.println(DASHES);
System.out.println("2. Getting " + userName + " in the user pool");
getAdminUser(identityProviderClient, userName, poolId);

System.out
    .println("*** Conformation code sent to " + userName + ". Would
you like to send a new code? (Yes/No)");
System.out.println(DASHES);

System.out.println(DASHES);
String ans = in.nextLine();

if (ans.compareTo("Yes") == 0) {
    resendConfirmationCode(identityProviderClient, clientId, userName);
    System.out.println("3. Sending a new confirmation code");
}
System.out.println(DASHES);

System.out.println(DASHES);
System.out.println("4. Enter confirmation code that was emailed");
String code = in.nextLine();
confirmSignUp(identityProviderClient, clientId, code, userName);
System.out.println("Rechecking the status of " + userName + " in the user
pool");
getAdminUser(identityProviderClient, userName, poolId);
System.out.println(DASHES);

System.out.println(DASHES);
System.out.println("5. Invokes the initiateAuth to sign in");
AdminInitiateAuthResponse authResponse =
initiateAuth(identityProviderClient, clientId, userName, password,
    poolId);
String mySession = authResponse.session();
System.out.println(DASHES);
```



```
        System.out.println(DASHES);
        System.out.println("6. Invokes the AssociateSoftwareToken method to
generate a TOTP key");
        String newSession = getSecretForAppMFA(identityProviderClient,
mySession);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("*** Enter the 6-digit code displayed in Google
Authenticator");
        String myCode = in.nextLine();
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("7. Verify the TOTP and register for MFA");
        verifyTOTP(identityProviderClient, newSession, myCode);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("8. Re-enter a 6-digit code displayed in Google
Authenticator");
        String mfaCode = in.nextLine();
        AdminInitiateAuthResponse authResponse1 =
initiateAuth(identityProviderClient, clientId, userName, password,
poolId);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("9. Invokes the AdminRespondToAuthChallenge");
        String session2 = authResponse1.session();
        adminRespondToAuthChallenge(identityProviderClient, userName, clientId,
mfaCode, session2);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("All Amazon Cognito operations were successfully
performed");
        System.out.println(DASHES);
    }

    // Respond to an authentication challenge.
    public static void adminRespondToAuthChallenge(CognitoIdentityProviderClient
identityProviderClient,
        String userName, String clientId, String mfaCode, String session) {
```

```
System.out.println("SOFTWARE_TOKEN_MFA challenge is generated");
Map<String, String> challengeResponses = new HashMap<>();

challengeResponses.put("USERNAME", userName);
challengeResponses.put("SOFTWARE_TOKEN_MFA_CODE", mfaCode);

AdminRespondToAuthChallengeRequest respondToAuthChallengeRequest =
AdminRespondToAuthChallengeRequest.builder()
    .challengeName(ChallengeNameType.SOFTWARE_TOKEN_MFA)
    .clientId(clientId)
    .challengeResponses(challengeResponses)
    .session(session)
    .build();

AdminRespondToAuthChallengeResponse respondToAuthChallengeResult =
identityProviderClient
    .adminRespondToAuthChallenge(respondToAuthChallengeRequest);

System.out.println("respondToAuthChallengeResult.getAuthenticationResult()"
    + respondToAuthChallengeResult.authenticationResult());
}

// Verify the TOTP and register for MFA.
public static void verifyTOTP(CognitoIdentityProviderClient
identityProviderClient, String session, String code) {
    try {
        VerifySoftwareTokenRequest tokenRequest =
VerifySoftwareTokenRequest.builder()
            .userCode(code)
            .session(session)
            .build();

        VerifySoftwareTokenResponse verifyResponse =
identityProviderClient.verifySoftwareToken(tokenRequest);
        System.out.println("The status of the token is " +
verifyResponse.statusAsString());

    } catch (CognitoIdentityProviderException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
```

```
public static AdminInitiateAuthResponse
initiateAuth(CognitoIdentityProviderClient identityProviderClient,
             String clientId, String userName, String password, String userPoolId)
{
    try {
        Map<String, String> authParameters = new HashMap<>();
        authParameters.put("USERNAME", userName);
        authParameters.put("PASSWORD", password);

        AdminInitiateAuthRequest authRequest =
AdminInitiateAuthRequest.builder()
            .clientId(clientId)
            .userPoolId(userPoolId)
            .authParameters(authParameters)
            .authFlow(AuthFlowType.ADMIN_USER_PASSWORD_AUTH)
            .build();

        AdminInitiateAuthResponse response =
identityProviderClient.adminInitiateAuth(authRequest);
        System.out.println("Result Challenge is : " +
response.challengeName());
        return response;

    } catch (CognitoIdentityProviderException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }

    return null;
}

public static String getSecretForAppMFA(CognitoIdentityProviderClient
identityProviderClient, String session) {
    AssociateSoftwareTokenRequest softwareTokenRequest =
AssociateSoftwareTokenRequest.builder()
        .session(session)
        .build();

    AssociateSoftwareTokenResponse tokenResponse = identityProviderClient
        .associateSoftwareToken(softwareTokenRequest);
    String secretCode = tokenResponse.secretCode();
    System.out.println("Enter this token into Google Authenticator");
    System.out.println(secretCode);
    return tokenResponse.session();
}
```

```
    }

    public static void confirmSignUp(CognitoIdentityProviderClient
identityProviderClient, String clientId, String code,
    String userName) {
        try {
            ConfirmSignUpRequest signUpRequest = ConfirmSignUpRequest.builder()
                .clientId(clientId)
                .confirmationCode(code)
                .username(userName)
                .build();

            identityProviderClient.confirmSignUp(signUpRequest);
            System.out.println(userName + " was confirmed");

        } catch (CognitoIdentityProviderException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void resendConfirmationCode(CognitoIdentityProviderClient
identityProviderClient, String clientId,
    String userName) {
        try {
            ResendConfirmationCodeRequest codeRequest =
ResendConfirmationCodeRequest.builder()
                .clientId(clientId)
                .username(userName)
                .build();

            ResendConfirmationCodeResponse response =
identityProviderClient.resendConfirmationCode(codeRequest);
            System.out.println("Method of delivery is " +
response.codeDeliveryDetails().deliveryMediumAsString());

        } catch (CognitoIdentityProviderException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void signUp(CognitoIdentityProviderClient
identityProviderClient, String clientId, String userName,
```

```
        String password, String email) {
    AttributeType userAttrs = AttributeType.builder()
        .name("email")
        .value(email)
        .build();

    List<AttributeType> userAttrsList = new ArrayList<>();
    userAttrsList.add(userAttrs);
    try {
        SignUpRequest signUpRequest = SignUpRequest.builder()
            .userAttributes(userAttrsList)
            .username(userName)
            .clientId(clientId)
            .password(password)
            .build();

        identityProviderClient.signUp(signUpRequest);
        System.out.println("User has been signed up ");

    } catch (CognitoIdentityProviderException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}

public static void getAdminUser(CognitoIdentityProviderClient
identityProviderClient, String userName,
    String poolId) {
    try {
        AdminGetUserRequest userRequest = AdminGetUserRequest.builder()
            .username(userName)
            .userPoolId(poolId)
            .build();

        AdminGetUserResponse response =
identityProviderClient.adminGetUser(userRequest);
        System.out.println("User status " + response.userStatusAsString());

    } catch (CognitoIdentityProviderException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
}
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK for Java 2.x .
  - [AdminGetUser](#)
  - [AdminInitiateAuth](#)
  - [AdminRespondToAuthChallenge](#)
  - [AssociateSoftwareToken](#)
  - [ConfirmDevice](#)
  - [ConfirmSignUp](#)
  - [InitiateAuth](#)
  - [ListUsers](#)
  - [ResendConfirmationCode](#)
  - [RespondToAuthChallenge](#)
  - [SignUp](#)
  - [VerifySoftwareToken](#)

## JavaScript

### SDK untuk JavaScript (v3)

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

Untuk pengalaman terbaik, kloning GitHub repositori dan jalankan contoh ini. Kode berikut merupakan contoh aplikasi contoh lengkap.

```
import { log } from "@aws-doc-sdk-examples/lib/utils/util-log.js";
import { signUp } from "../../actions/sign-up.js";
import { FILE_USER_POOLS } from "./constants.js";
import { getSecondValuesFromEntries } from "@aws-doc-sdk-examples/lib/utils/util-csv.js";

const validateClient = (clientId) => {
```

```
    if (!clientId) {
      throw new Error(
        `App client id is missing. Did you run 'create-user-pool'?`,
      );
    }
  };

const validateUser = (username, password, email) => {
  if (!(username && password && email)) {
    throw new Error(
      `Username, password, and email must be provided as arguments to the 'sign-up' command.`,
    );
  }
};

const signUpHandler = async (commands) => {
  const [, username, password, email] = commands;

  try {
    validateUser(username, password, email);
    /**
     * @type {string[]}
     */
    const values = getSecondValuesFromEntries(FILE_USER_POOLS);
    const clientId = values[0];
    validateClient(clientId);
    log(`Signing up.`);
    await signUp({ clientId, username, password, email });
    log(`Signed up. A confirmation email has been sent to: ${email}.`);
    log(`Run 'confirm-sign-up ${username} <code>' to confirm your account.`);
  } catch (err) {
    log(err);
  }
};

export { signUpHandler };

const signUp = ({ clientId, username, password, email }) => {
  const client = new CognitoIdentityProviderClient({});

  const command = new SignUpCommand({
    ClientId: clientId,
    Username: username,
```

```
    Password: password,
    UserAttributes: [{ Name: "email", Value: email }],
  });

  return client.send(command);
};

import { log } from "@aws-doc-sdk-examples/lib/utils/util-log.js";
import { confirmSignUp } from "../../actions/confirm-sign-up.js";
import { FILE_USER_POOLS } from "./constants.js";
import { getSecondValuesFromEntries } from "@aws-doc-sdk-examples/lib/utils/util-csv.js";

const validateClient = (clientId) => {
  if (!clientId) {
    throw new Error(
      `App client id is missing. Did you run 'create-user-pool'?`,
    );
  }
};

const validateUser = (username) => {
  if (!username) {
    throw new Error(
      `Username name is missing. It must be provided as an argument to the 'confirm-sign-up' command.`,
    );
  }
};

const validateCode = (code) => {
  if (!code) {
    throw new Error(
      `Verification code is missing. It must be provided as an argument to the 'confirm-sign-up' command.`,
    );
  }
};

const confirmSignUpHandler = async (commands) => {
  const [, username, code] = commands;

  try {
    validateUser(username);
```



```
validateCode(code);
/**
 * @type {string[]}
 */
const values = getSecondValuesFromEntries(FILE_USER_POOLS);
const clientId = values[0];
validateClient(clientId);
log(`Confirming user.`);
await confirmSignUp({ clientId, username, code });
log(
  `User confirmed. Run 'admin-initiate-auth ${username} <password>' to sign
in.`,
);
} catch (err) {
  log(err);
}
};

export { confirmSignUpHandler };

const confirmSignUp = ({ clientId, username, code }) => {
  const client = new CognitoIdentityProviderClient({});

  const command = new ConfirmSignUpCommand({
    ClientId: clientId,
    Username: username,
    ConfirmationCode: code,
  });

  return client.send(command);
};

import qrCode from "qr-code-terminal";
import { log } from "@aws-doc-sdk-examples/lib/utils/util-log.js";
import { adminInitiateAuth } from "../../actions/admin-initiate-auth.js";
import { associateSoftwareToken } from "../../actions/associate-software-token.js";
import { FILE_USER_POOLS } from "../constants.js";
import { getFirstEntry } from "@aws-doc-sdk-examples/lib/utils/util-csv.js";

const handleMfaSetup = async (session, username) => {
  const { SecretCode, Session } = await associateSoftwareToken(session);

  // Store the Session for use with 'VerifySoftwareToken'.
```

```
process.env.SESSION = Session;

console.log(
  "Scan this code in your preferred authenticator app, then run 'verify-
software-token' to finish the setup.",
);
qrcode.generate(
  `otpauth://totp/${username}?secret=${SecretCode}`,
  { small: true },
  console.log,
);
};

const handleSoftwareTokenMfa = (session) => {
  // Store the Session for use with 'AdminRespondToAuthChallenge'.
  process.env.SESSION = session;
};

const validateClient = (id) => {
  if (!id) {
    throw new Error(
      `User pool client id is missing. Did you run 'create-user-pool'?`,
    );
  }
};

const validateId = (id) => {
  if (!id) {
    throw new Error(`User pool id is missing. Did you run 'create-user-pool'?`);
  }
};

const validateUser = (username, password) => {
  if (!(username && password)) {
    throw new Error(
      `Username and password must be provided as arguments to the 'admin-
initiate-auth' command.`,
    );
  }
};

const adminInitiateAuthHandler = async (commands) => {
  const [, username, password] = commands;
```

```
try {
  validateUser(username, password);

  const [userPoolId, clientId] = getFirstEntry(FILE_USER_POOLS);
  validateId(userPoolId);
  validateClient(clientId);

  log("Signing in.");
  const { ChallengeName, Session } = await adminInitiateAuth({
    clientId,
    userPoolId,
    username,
    password,
  });

  if (ChallengeName === "MFA_SETUP") {
    log("MFA setup is required.");
    return handleMfaSetup(Session, username);
  }

  if (ChallengeName === "SOFTWARE_TOKEN_MFA") {
    handleSoftwareTokenMfa(Session);
    log(`Run 'admin-respond-to-auth-challenge ${username} <totp>'`);
  }
} catch (err) {
  log(err);
}

export { adminInitiateAuthHandler };

const adminInitiateAuth = ({ clientId, userPoolId, username, password }) => {
  const client = new CognitoIdentityProviderClient({});

  const command = new AdminInitiateAuthCommand({
    ClientId: clientId,
    UserPoolId: userPoolId,
    AuthFlow: AuthFlowType.ADMIN_USER_PASSWORD_AUTH,
    AuthParameters: { USERNAME: username, PASSWORD: password },
  });

  return client.send(command);
};
```

```
import { log } from "@aws-doc-sdk-examples/lib/utils/util-log.js";
import { adminRespondToAuthChallenge } from "../../actions/admin-respond-to-auth-challenge.js";
import { getFirstEntry } from "@aws-doc-sdk-examples/lib/utils/util-csv.js";
import { FILE_USER_POOLS } from "./constants.js";

const verifyUsername = (username) => {
  if (!username) {
    throw new Error(
      `Username is missing. It must be provided as an argument to the 'admin-respond-to-auth-challenge' command.`
    );
  }
};

const verifyTotp = (totp) => {
  if (!totp) {
    throw new Error(
      `Time-based one-time password (TOTP) is missing. It must be provided as an argument to the 'admin-respond-to-auth-challenge' command.`
    );
  }
};

const storeAccessToken = (token) => {
  process.env.AccessToken = token;
};

const adminRespondToAuthChallengeHandler = async (commands) => {
  const [, username, totp] = commands;

  try {
    verifyUsername(username);
    verifyTotp(totp);

    const [userPoolId, clientId] = getFirstEntry(FILE_USER_POOLS);
    const session = process.env.SESSION;

    const { AuthenticationResult } = await adminRespondToAuthChallenge({
      clientId,
      userPoolId,
      username,
      totp,
      session,
    });
  }
};
```

```
});

storeAccessToken(AuthenticationResult.AccessToken);

log("Successfully authenticated.");
} catch (err) {
  log(err);
}
};

export { adminRespondToAuthChallengeHandler };

const respondToAuthChallenge = ({
  clientId,
  username,
  session,
  userPoolId,
  code,
}) => {
  const client = new CognitoIdentityProviderClient({});

  const command = new RespondToAuthChallengeCommand({
    ChallengeName: ChallengeNameType.SOFTWARE_TOKEN_MFA,
    ChallengeResponses: {
      SOFTWARE_TOKEN_MFA_CODE: code,
      USERNAME: username,
    },
    ClientId: clientId,
    UserPoolId: userPoolId,
    Session: session,
  });

  return client.send(command);
};

import { log } from "@aws-doc-sdk-examples/lib/utils/util-log.js";
import { verifySoftwareToken } from "../../actions/verify-software-token.js";

const validateTotp = (totp) => {
  if (!totp) {
    throw new Error(
      `Time-based one-time password (TOTP) must be provided to the 'validate-software-token' command.`
    );
  }
};
```

```
    }  
  };  
  const verifySoftwareTokenHandler = async (commands) => {  
    const [_ , totp] = commands;  
  
    try {  
      validateTotp(totp);  
  
      log("Verifying TOTP.");  
      await verifySoftwareToken(totp);  
      log("TOTP Verified. Run 'admin-initiate-auth' again to sign-in.");  
    } catch (err) {  
      console.log(err);  
    }  
  };  
  
  export { verifySoftwareTokenHandler };  
  
  const verifySoftwareToken = (totp) => {  
    const client = new CognitoIdentityProviderClient({});  
  
    // The 'Session' is provided in the response to 'AssociateSoftwareToken'.  
    const session = process.env.SESSION;  
  
    if (!session) {  
      throw new Error(  
        "Missing a valid Session. Did you run 'admin-initiate-auth'?",  
      );  
    }  
  
    const command = new VerifySoftwareTokenCommand({  
      Session: session,  
      UserCode: totp,  
    });  
  
    return client.send(command);  
  };  
};
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK for JavaScript .
  - [AdminGetUser](#)
  - [AdminInitiateAuth](#)

- [AdminRespondToAuthChallenge](#)
- [AssociateSoftwareToken](#)
- [ConfirmDevice](#)
- [ConfirmSignUp](#)
- [InitiateAuth](#)
- [ListUsers](#)
- [ResendConfirmationCode](#)
- [RespondToAuthChallenge](#)
- [SignUp](#)
- [VerifySoftwareToken](#)

## Kotlin

### SDK untuk Kotlin

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkap dan pelajari cara menyiapkan dan menjalankan di [Repositori Contoh Kode AWS](#).

```
/**
```

```
Before running this Kotlin code example, set up your development environment, including your credentials.
```

```
For more information, see the following documentation:
```

```
https://docs.aws.amazon.com/sdk-for-kotlin/latest/developer-guide/setup.html
```

```
TIP: To set up the required user pool, run the AWS Cloud Development Kit (AWS CDK) script provided in this GitHub repo at resources/cdk/cognito_scenario_user_pool_with_mfa.
```

```
This code example performs the following operations:
```

1. Invokes the `signUp` method to sign up a user.
2. Invokes the `adminGetUser` method to get the user's confirmation status.
3. Invokes the `ResendConfirmationCode` method if the user requested another code.

4. Invokes the `confirmSignUp` method.
  5. Invokes the `initiateAuth` to sign in. This results in being prompted to set up TOTP (time-based one-time password). (The response is `"ChallengeName": "MFA_SETUP"`).
  6. Invokes the `AssociateSoftwareToken` method to generate a TOTP MFA private key. This can be used with Google Authenticator.
  7. Invokes the `VerifySoftwareToken` method to verify the TOTP and register for MFA.
  8. Invokes the `AdminInitiateAuth` to sign in again. This results in being prompted to submit a TOTP (Response: `"ChallengeName": "SOFTWARE_TOKEN_MFA"`).
  9. Invokes the `AdminRespondToAuthChallenge` to get back a token.
- \*/

```
suspend fun main(args: Array<String>) {
    val usage = """
        Usage:
            <clientId> <poolId>
        Where:
            clientId - The app client Id value that you can get from the AWS CDK
script.
            poolId - The pool Id that you can get from the AWS CDK script.
        """

    if (args.size != 2) {
        println(usage)
        exitProcess(1)
    }

    val clientId = args[0]
    val poolId = args[1]

    // Use the console to get data from the user.
    println("*** Enter your use name")
    val in0b = Scanner(System.`in`)
    val userName = in0b.nextLine()
    println(userName)

    println("*** Enter your password")
    val password: String = in0b.nextLine()

    println("*** Enter your email")
    val email = in0b.nextLine()

    println("*** Signing up $userName")
```



```
signUp(clientId, userName, password, email)

println("**** Getting $userName in the user pool")
getAdminUser(userName, poolId)

println("**** Conformation code sent to $userName. Would you like to send a
new code? (Yes/No)")
val ans = in0b.nextLine()

if (ans.compareTo("Yes") == 0) {
    println("**** Sending a new confirmation code")
    resendConfirmationCode(clientId, userName)
}
println("**** Enter the confirmation code that was emailed")
val code = in0b.nextLine()
confirmSignUp(clientId, code, userName)

println("**** Rechecking the status of $userName in the user pool")
getAdminUser(userName, poolId)

val authResponse = checkAuthMethod(clientId, userName, password, poolId)
val mySession = authResponse.session
val newSession = getSecretForAppMFA(mySession)
println("**** Enter the 6-digit code displayed in Google Authenticator")
val myCode = in0b.nextLine()

// Verify the TOTP and register for MFA.
verifyTOTP(newSession, myCode)
println("**** Re-enter a 6-digit code displayed in Google Authenticator")
val mfaCode: String = in0b.nextLine()
val authResponse1 = checkAuthMethod(clientId, userName, password, poolId)
val session2 = authResponse1.session
adminRespondToAuthChallenge(userName, clientId, mfaCode, session2)
}

suspend fun checkAuthMethod(clientIdVal: String, userNameVal: String,
passwordVal: String, userPoolIdVal: String): AdminInitiateAuthResponse {
    val authParas = mutableMapOf<String, String>()
    authParas["USERNAME"] = userNameVal
    authParas["PASSWORD"] = passwordVal

    val authRequest = AdminInitiateAuthRequest {
        clientId = clientIdVal
        userPoolId = userPoolIdVal
```

```
        authParameters = authParas
        authFlow = AuthFlowType.AdminUserPasswordAuth
    }

    CognitoIdentityProviderClient { region = "us-east-1" }.use
{ identityProviderClient ->
    val response = identityProviderClient.adminInitiateAuth(authRequest)
    println("Result Challenge is ${response.challengeName}")
    return response
}
}

suspend fun resendConfirmationCode(clientIdVal: String?, userNameVal: String?) {
    val codeRequest = ResendConfirmationCodeRequest {
        clientId = clientIdVal
        username = userNameVal
    }

    CognitoIdentityProviderClient { region = "us-east-1" }.use
{ identityProviderClient ->
    val response = identityProviderClient.resendConfirmationCode(codeRequest)
    println("Method of delivery is " +
(response.codeDeliveryDetails?.deliveryMedium))
    }
}

// Respond to an authentication challenge.
suspend fun adminRespondToAuthChallenge(userName: String, clientIdVal: String?,
mfaCode: String, sessionVal: String?) {
    println("SOFTWARE_TOKEN_MFA challenge is generated")
    val challengeResponsesOb = mutableMapOf<String, String>()
    challengeResponsesOb["USERNAME"] = userName
    challengeResponsesOb["SOFTWARE_TOKEN_MFA_CODE"] = mfaCode

    val adminRespondToAuthChallengeRequest = AdminRespondToAuthChallengeRequest {
        challengeName = ChallengeNameType.SoftwareTokenMfa
        clientId = clientIdVal
        challengeResponses = challengeResponsesOb
        session = sessionVal
    }

    CognitoIdentityProviderClient { region = "us-east-1" }.use
{ identityProviderClient ->
```

```
        val respondToAuthChallengeResult =
identityProviderClient.adminRespondToAuthChallenge(adminRespondToAuthChallengeRequest)
        println("respondToAuthChallengeResult.getAuthenticationResult()
${respondToAuthChallengeResult.authenticationResult}")
    }
}

// Verify the TOTP and register for MFA.
suspend fun verifyTOTP(sessionVal: String?, codeVal: String?) {
    val tokenRequest = VerifySoftwareTokenRequest {
        userCode = codeVal
        session = sessionVal
    }

    CognitoIdentityProviderClient { region = "us-east-1" }.use
{ identityProviderClient ->
    val verifyResponse =
identityProviderClient.verifySoftwareToken(tokenRequest)
    println("The status of the token is ${verifyResponse.status}")
}
}

suspend fun getSecretForAppMFA(sessionVal: String?): String? {
    val softwareTokenRequest = AssociateSoftwareTokenRequest {
        session = sessionVal
    }

    CognitoIdentityProviderClient { region = "us-east-1" }.use
{ identityProviderClient ->
    val tokenResponse =
identityProviderClient.associateSoftwareToken(softwareTokenRequest)
    val secretCode = tokenResponse.secretCode
    println("Enter this token into Google Authenticator")
    println(secretCode)
    return tokenResponse.session
}
}

suspend fun confirmSignUp(clientIdVal: String?, codeVal: String?, userNameVal:
String?) {
    val signUpRequest = ConfirmSignUpRequest {
        clientId = clientIdVal
        confirmationCode = codeVal
        username = userNameVal
    }
}
```

```
    }

    CognitoIdentityProviderClient { region = "us-east-1" }.use
{ identityProviderClient ->
    identityProviderClient.confirmSignUp(signUpRequest)
    println("$userNameVal was confirmed")
}
}

suspend fun getAdminUser(userNameVal: String?, poolIdVal: String?) {
    val userRequest = AdminGetUserRequest {
        username = userNameVal
        userPoolId = poolIdVal
    }

    CognitoIdentityProviderClient { region = "us-east-1" }.use
{ identityProviderClient ->
    val response = identityProviderClient.adminGetUser(userRequest)
    println("User status ${response.userStatus}")
}
}

suspend fun signUp(clientIdVal: String?, userNameVal: String?, passwordVal:
String?, emailVal: String?) {
    val userAttrs = AttributeType {
        name = "email"
        value = emailVal
    }

    val userAttrsList = mutableListof<AttributeType>()
    userAttrsList.add(userAttrs)
    val signUpRequest = SignUpRequest {
        userAttributes = userAttrsList
        username = userNameVal
        clientId = clientIdVal
        password = passwordVal
    }

    CognitoIdentityProviderClient { region = "us-east-1" }.use
{ identityProviderClient ->
    identityProviderClient.signUp(signUpRequest)
    println("User has been signed up")
}
}
```

- Untuk detail API, lihat topik berikut di referensi API SDK untuk Kotlin AWS .
  - [AdminGetUser](#)
  - [AdminInitiateAuth](#)
  - [AdminRespondToAuthChallenge](#)
  - [AssociateSoftwareToken](#)
  - [ConfirmDevice](#)
  - [ConfirmSignUp](#)
  - [InitiateAuth](#)
  - [ListUsers](#)
  - [ResendConfirmationCode](#)
  - [RespondToAuthChallenge](#)
  - [SignUp](#)
  - [VerifySoftwareToken](#)

## Python

### SDK untuk Python (Boto3)

#### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

Buat kelas yang membungkus fungsi Amazon Cognito yang digunakan dalam skenario.

```
class CognitoIdentityProviderWrapper:
    """Encapsulates Amazon Cognito actions"""

    def __init__(self, cognito_idp_client, user_pool_id, client_id,
                 client_secret=None):
        """
        :param cognito_idp_client: A Boto3 Amazon Cognito Identity Provider
        client.
        :param user_pool_id: The ID of an existing Amazon Cognito user pool.
```

```
pool.  
    :param client_id: The ID of a client application registered with the user  
    """  
    :param client_secret: The client secret, if the client has a secret.  
    """  
    self.cognito_idp_client = cognito_idp_client  
    self.user_pool_id = user_pool_id  
    self.client_id = client_id  
    self.client_secret = client_secret  
  
def _secret_hash(self, user_name):  
    """  
    Calculates a secret hash from a user name and a client secret.  
  
    :param user_name: The user name to use when calculating the hash.  
    :return: The secret hash.  
    """  
    key = self.client_secret.encode()  
    msg = bytes(user_name + self.client_id, "utf-8")  
    secret_hash = base64.b64encode(  
        hmac.new(key, msg, digestmod=hashlib.sha256).digest()  
    ).decode()  
    logger.info("Made secret hash for %s: %s.", user_name, secret_hash)  
    return secret_hash  
  
def sign_up_user(self, user_name, password, user_email):  
    """  
    Signs up a new user with Amazon Cognito. This action prompts Amazon  
Cognito  
    to send an email to the specified email address. The email contains a  
code that  
    can be used to confirm the user.  
  
    When the user already exists, the user status is checked to determine  
whether  
    the user has been confirmed.  
  
    :param user_name: The user name that identifies the new user.  
    :param password: The password for the new user.  
    :param user_email: The email address for the new user.  
    :return: True when the user is already confirmed with Amazon Cognito.  
            Otherwise, false.  
    """  
    try:
```

```
        kwargs = {
            "ClientId": self.client_id,
            "Username": user_name,
            "Password": password,
            "UserAttributes": [{"Name": "email", "Value": user_email}],
        }
        if self.client_secret is not None:
            kwargs["SecretHash"] = self._secret_hash(user_name)
        response = self.cognito_idp_client.sign_up(**kwargs)
        confirmed = response["UserConfirmed"]
    except ClientError as err:
        if err.response["Error"]["Code"] == "UsernameExistsException":
            response = self.cognito_idp_client.admin_get_user(
                UserPoolId=self.user_pool_id, Username=user_name
            )
            logger.warning(
                "User %s exists and is %s.", user_name,
                response["UserStatus"]
            )
            confirmed = response["UserStatus"] == "CONFIRMED"
        else:
            logger.error(
                "Couldn't sign up %s. Here's why: %s: %s",
                user_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
    return confirmed

def resend_confirmation(self, user_name):
    """
    Prompts Amazon Cognito to resend an email with a new confirmation code.

    :param user_name: The name of the user who will receive the email.
    :return: Delivery information about where the email is sent.
    """
    try:
        kwargs = {"ClientId": self.client_id, "Username": user_name}
        if self.client_secret is not None:
            kwargs["SecretHash"] = self._secret_hash(user_name)
        response = self.cognito_idp_client.resend_confirmation_code(**kwargs)
        delivery = response["CodeDeliveryDetails"]
```

```
except ClientError as err:
    logger.error(
        "Couldn't resend confirmation to %s. Here's why: %s: %s",
        user_name,
        err.response["Error"]["Code"],
        err.response["Error"]["Message"],
    )
    raise
else:
    return delivery

def confirm_user_sign_up(self, user_name, confirmation_code):
    """
    Confirms a previously created user. A user must be confirmed before they
    can sign in to Amazon Cognito.

    :param user_name: The name of the user to confirm.
    :param confirmation_code: The confirmation code sent to the user's
registered
                               email address.
    :return: True when the confirmation succeeds.
    """
    try:
        kwargs = {
            "ClientId": self.client_id,
            "Username": user_name,
            "ConfirmationCode": confirmation_code,
        }
        if self.client_secret is not None:
            kwargs["SecretHash"] = self._secret_hash(user_name)
        self.cognito_idp_client.confirm_sign_up(**kwargs)
    except ClientError as err:
        logger.error(
            "Couldn't confirm sign up for %s. Here's why: %s: %s",
            user_name,
            err.response["Error"]["Code"],
            err.response["Error"]["Message"],
        )
        raise
    else:
        return True
```



```
def list_users(self):
    """
    Returns a list of the users in the current user pool.

    :return: The list of users.
    """
    try:
        response =
self.cognito_idp_client.list_users(UserPoolId=self.user_pool_id)
        users = response["Users"]
    except ClientError as err:
        logger.error(
            "Couldn't list users for %s. Here's why: %s: %s",
            self.user_pool_id,
            err.response["Error"]["Code"],
            err.response["Error"]["Message"],
        )
        raise
    else:
        return users

def start_sign_in(self, user_name, password):
    """
    Starts the sign-in process for a user by using administrator credentials.
    This method of signing in is appropriate for code running on a secure
server.

    If the user pool is configured to require MFA and this is the first sign-
in
    for the user, Amazon Cognito returns a challenge response to set up an
MFA application. When this occurs, this function gets an MFA secret from
Amazon Cognito and returns it to the caller.

    :param user_name: The name of the user to sign in.
    :param password: The user's password.
    :return: The result of the sign-in attempt. When sign-in is successful,
this
        returns an access token that can be used to get AWS credentials.
    Otherwise,
        Amazon Cognito returns a challenge to set up an MFA application,
or a challenge to enter an MFA code from a registered MFA
application.
    """
```

```
try:
    kwargs = {
        "UserPoolId": self.user_pool_id,
        "ClientId": self.client_id,
        "AuthFlow": "ADMIN_USER_PASSWORD_AUTH",
        "AuthParameters": {"USERNAME": user_name, "PASSWORD": password},
    }
    if self.client_secret is not None:
        kwargs["AuthParameters"]["SECRET_HASH"] =
self._secret_hash(user_name)
    response = self.cognito_idp_client.admin_initiate_auth(**kwargs)
    challenge_name = response.get("ChallengeName", None)
    if challenge_name == "MFA_SETUP":
        if (
            "SOFTWARE_TOKEN_MFA"
            in response["ChallengeParameters"]["MFAS_CAN_SETUP"]
        ):
            response.update(self.get_mfa_secret(response["Session"]))
        else:
            raise RuntimeError(
                "The user pool requires MFA setup, but the user pool is
not "
                "configured for TOTP MFA. This example requires TOTP
MFA."
            )
    except ClientError as err:
        logger.error(
            "Couldn't start sign in for %s. Here's why: %s: %s",
            user_name,
            err.response["Error"]["Code"],
            err.response["Error"]["Message"],
        )
        raise
    else:
        response.pop("ResponseMetadata", None)
        return response

def get_mfa_secret(self, session):
    """
    Gets a token that can be used to associate an MFA application with the
user.
```

```
        :param session: Session information returned from a previous call to
initiate
                authentication.
        :return: An MFA token that can be used to set up an MFA application.
        """
        try:
            response =
self.cognito_idp_client.associate_software_token(Session=session)
        except ClientError as err:
            logger.error(
                "Couldn't get MFA secret. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            response.pop("ResponseMetadata", None)
            return response

def verify_mfa(self, session, user_code):
    """
    Verify a new MFA application that is associated with a user.

    :param session: Session information returned from a previous call to
initiate
                authentication.
    :param user_code: A code generated by the associated MFA application.
    :return: Status that indicates whether the MFA application is verified.
    """
    try:
        response = self.cognito_idp_client.verify_software_token(
            Session=session, UserCode=user_code
        )
    except ClientError as err:
        logger.error(
            "Couldn't verify MFA. Here's why: %s: %s",
            err.response["Error"]["Code"],
            err.response["Error"]["Message"],
        )
        raise
    else:
        response.pop("ResponseMetadata", None)
        return response
```

```
def respond_to_mfa_challenge(self, user_name, session, mfa_code):
    """
    Responds to a challenge for an MFA code. This completes the second step
of
a two-factor sign-in. When sign-in is successful, it returns an access
token
that can be used to get AWS credentials from Amazon Cognito.

:param user_name: The name of the user who is signing in.
:param session: Session information returned from a previous call to
initiate
authentication.
:param mfa_code: A code generated by the associated MFA application.
:return: The result of the authentication. When successful, this contains
an
access token for the user.
    """
    try:
        kwargs = {
            "UserPoolId": self.user_pool_id,
            "ClientId": self.client_id,
            "ChallengeName": "SOFTWARE_TOKEN_MFA",
            "Session": session,
            "ChallengeResponses": {
                "USERNAME": user_name,
                "SOFTWARE_TOKEN_MFA_CODE": mfa_code,
            },
        }
        if self.client_secret is not None:
            kwargs["ChallengeResponses"]["SECRET_HASH"] = self._secret_hash(
                user_name
            )
        response =
self.cognito_idp_client.admin_respond_to_auth_challenge(**kwargs)
        auth_result = response["AuthenticationResult"]
    except ClientError as err:
        if err.response["Error"]["Code"] == "ExpiredCodeException":
            logger.warning(
                "Your MFA code has expired or has been used already. You
might have "
                "to wait a few seconds until your app shows you a new code."
            )
```

```

        else:
            logger.error(
                "Couldn't respond to mfa challenge for %s. Here's why: %s:
%s",
                user_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
    else:
        return auth_result

def confirm_mfa_device(
    self,
    user_name,
    device_key,
    device_group_key,
    device_password,
    access_token,
    aws_srp,
):
    """
    Confirms an MFA device to be tracked by Amazon Cognito. When a device is
    tracked, its key and password can be used to sign in without requiring a
    new
    MFA code from the MFA application.

    :param user_name: The user that is associated with the device.
    :param device_key: The key of the device, returned by Amazon Cognito.
    :param device_group_key: The group key of the device, returned by Amazon
    Cognito.
    :param device_password: The password that is associated with the device.
    :param access_token: The user's access token.
    :param aws_srp: A class that helps with Secure Remote Password (SRP)
    calculations. The scenario associated with this example
    uses
        the warrant package.
    :return: True when the user must confirm the device. Otherwise, False.
    When
        False, the device is automatically confirmed and tracked.
    """
    srp_helper = aws_srp.AWSSRP(
        username=user_name,

```

```
        password=device_password,
        pool_id="",
        client_id=self.client_id,
        client_secret=None,
        client=self.cognito_idp_client,
    )
    device_and_pw = f"{device_group_key}{device_key}:{device_password}"
    device_and_pw_hash = aws_srp.hash_sha256(device_and_pw.encode("utf-8"))
    salt = aws_srp.pad_hex(aws_srp.get_random(16))
    x_value = aws_srp.hex_to_long(aws_srp.hex_hash(salt +
device_and_pw_hash))
    verifier = aws_srp.pad_hex(pow(srp_helper.val_g, x_value,
srp_helper.big_n))
    device_secret_verifier_config = {
        "PasswordVerifier": base64.standard_b64encode(
            bytearray.fromhex(verifier)
        ).decode("utf-8"),
        "Salt":
base64.standard_b64encode(bytearray.fromhex(salt)).decode("utf-8"),
    }
    try:
        response = self.cognito_idp_client.confirm_device(
            AccessToken=access_token,
            DeviceKey=device_key,
            DeviceSecretVerifierConfig=device_secret_verifier_config,
        )
        user_confirm = response["UserConfirmationNecessary"]
    except ClientError as err:
        logger.error(
            "Couldn't confirm mfa device %s. Here's why: %s: %s",
            device_key,
            err.response["Error"]["Code"],
            err.response["Error"]["Message"],
        )
        raise
    else:
        return user_confirm

def sign_in_with_tracked_device(
    self,
    user_name,
    password,
    device_key,
```

```

        device_group_key,
        device_password,
        aws_srp,
    ):
        """
        Signs in to Amazon Cognito as a user who has a tracked device. Signing in
        with a tracked device lets a user sign in without entering a new MFA
code.

        Signing in with a tracked device requires that the client respond to the
SRP
        protocol. The scenario associated with this example uses the warrant
package
        to help with SRP calculations.

        For more information on SRP, see https://en.wikipedia.org/wiki/Secure\_Remote\_Password\_protocol.

        :param user_name: The user that is associated with the device.
        :param password: The user's password.
        :param device_key: The key of a tracked device.
        :param device_group_key: The group key of a tracked device.
        :param device_password: The password that is associated with the device.
        :param aws_srp: A class that helps with SRP calculations. The scenario
            associated with this example uses the warrant package.
        :return: The result of the authentication. When successful, this contains
an
            access token for the user.
        """
        try:
            srp_helper = aws_srp.AWSSRP(
                username=user_name,
                password=device_password,
                pool_id="",
                client_id=self.client_id,
                client_secret=None,
                client=self.cognito_idp_client,
            )

            response_init = self.cognito_idp_client.initiate_auth(
                ClientId=self.client_id,
                AuthFlow="USER_PASSWORD_AUTH",
                AuthParameters={
                    "USERNAME": user_name,

```

```
        "PASSWORD": password,
        "DEVICE_KEY": device_key,
    },
)
if response_init["ChallengeName"] != "DEVICE_SRP_AUTH":
    raise RuntimeError(
        f"Expected DEVICE_SRP_AUTH challenge but got
{response_init['ChallengeName']}."
    )

auth_params = srp_helper.get_auth_params()
auth_params["DEVICE_KEY"] = device_key
response_auth = self.cognito_idp_client.respond_to_auth_challenge(
    ClientId=self.client_id,
    ChallengeName="DEVICE_SRP_AUTH",
    ChallengeResponses=auth_params,
)
if response_auth["ChallengeName"] != "DEVICE_PASSWORD_VERIFIER":
    raise RuntimeError(
        f"Expected DEVICE_PASSWORD_VERIFIER challenge but got "
        f"{response_init['ChallengeName']}."
    )

challenge_params = response_auth["ChallengeParameters"]
challenge_params["USER_ID_FOR_SRP"] = device_group_key + device_key
cr = srp_helper.process_challenge(challenge_params, {"USERNAME":
user_name})
cr["USERNAME"] = user_name
cr["DEVICE_KEY"] = device_key
response_verifier =
self.cognito_idp_client.respond_to_auth_challenge(
    ClientId=self.client_id,
    ChallengeName="DEVICE_PASSWORD_VERIFIER",
    ChallengeResponses=cr,
)
auth_tokens = response_verifier["AuthenticationResult"]
except ClientError as err:
    logger.error(
        "Couldn't start client sign in for %s. Here's why: %s: %s",
        user_name,
        err.response["Error"]["Code"],
        err.response["Error"]["Message"],
    )
raise
```



```
else:
    return auth_tokens
```

Buat kelas yang menjalankan skenario. Contoh ini juga mendaftarkan perangkat MFA untuk dilacak oleh Amazon Cognito dan menunjukkan cara masuk dengan menggunakan kata sandi dan informasi dari perangkat yang dilacak. Ini menghindari kebutuhan untuk memasukkan kode MFA baru.

```
def run_scenario(cognito_idp_client, user_pool_id, client_id):
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")

    print("-" * 88)
    print("Welcome to the Amazon Cognito user signup with MFA demo.")
    print("-" * 88)

    cog_wrapper = CognitoIdentityProviderWrapper(
        cognito_idp_client, user_pool_id, client_id
    )

    user_name = q.ask("Let's sign up a new user. Enter a user name: ",
q.non_empty)
    password = q.ask("Enter a password for the user: ", q.non_empty)
    email = q.ask("Enter a valid email address that you own: ", q.non_empty)
    confirmed = cog_wrapper.sign_up_user(user_name, password, email)
    while not confirmed:
        print(
            f"User {user_name} requires confirmation. Check {email} for "
            f"a verification code."
        )
        confirmation_code = q.ask("Enter the confirmation code from the email: ")
        if not confirmation_code:
            if q.ask("Do you need another confirmation code (y/n)? ",
q.is_yesno):
                delivery = cog_wrapper.resend_confirmation(user_name)
                print(
                    f"Confirmation code sent by {delivery['DeliveryMedium']} "
                    f"to {delivery['Destination']}."
                )
            else:
```

```
        confirmed = cog_wrapper.confirm_user_sign_up(user_name,
confirmation_code)
    print(f"User {user_name} is confirmed and ready to use.")
    print("-" * 88)

    print("Let's get a list of users in the user pool.")
    q.ask("Press Enter when you're ready.")
    users = cog_wrapper.list_users()
    if users:
        print(f"Found {len(users)} users:")
        pp(users)
    else:
        print("No users found.")
    print("-" * 88)

    print("Let's sign in and get an access token.")
    auth_tokens = None
    challenge = "ADMIN_USER_PASSWORD_AUTH"
    response = {}
    while challenge is not None:
        if challenge == "ADMIN_USER_PASSWORD_AUTH":
            response = cog_wrapper.start_sign_in(user_name, password)
            challenge = response["ChallengeName"]
        elif response["ChallengeName"] == "MFA_SETUP":
            print("First, we need to set up an MFA application.")
            qr_img = qrcode.make(
                f"otpauth://totp/{user_name}?secret={response['SecretCode']}"
            )
            qr_img.save("qr.png")
            q.ask(
                "Press Enter to see a QR code on your screen. Scan it into an MFA
"
                "application, such as Google Authenticator."
            )
            webbrowser.open("qr.png")
            mfa_code = q.ask(
                "Enter the verification code from your MFA application: ",
q.non_empty
            )
            response = cog_wrapper.verify_mfa(response["Session"], mfa_code)
            print(f"MFA device setup {response['Status']}")
            print("Now that an MFA application is set up, let's sign in again.")
            print(
```

```
        "You might have to wait a few seconds for a new MFA code to
appear in "
        "your MFA application."
    )
    challenge = "ADMIN_USER_PASSWORD_AUTH"
    elif response["ChallengeName"] == "SOFTWARE_TOKEN_MFA":
        auth_tokens = None
        while auth_tokens is None:
            mfa_code = q.ask(
                "Enter a verification code from your MFA application: ",
q.non_empty
            )
            auth_tokens = cog_wrapper.respond_to_mfa_challenge(
                user_name, response["Session"], mfa_code
            )
            print(f"You're signed in as {user_name}.")
            print("Here's your access token:")
            pp(auth_tokens["AccessToken"])
            print("And your device information:")
            pp(auth_tokens["NewDeviceMetadata"])
            challenge = None
        else:
            raise Exception(f"Got unexpected challenge
{response['ChallengeName']}")
            print("-" * 88)

            device_group_key = auth_tokens["NewDeviceMetadata"]["DeviceGroupKey"]
            device_key = auth_tokens["NewDeviceMetadata"]["DeviceKey"]
            device_password = base64.standard_b64encode(os.urandom(40)).decode("utf-8")

            print("Let's confirm your MFA device so you don't have re-enter MFA tokens
for it.")
            q.ask("Press Enter when you're ready.")
            cog_wrapper.confirm_mfa_device(
                user_name,
                device_key,
                device_group_key,
                device_password,
                auth_tokens["AccessToken"],
                aws_srp,
            )
            print(f"Your device {device_key} is confirmed.")
            print("-" * 88)
```

```
print(
    f"Now let's sign in as {user_name} from your confirmed device
{device_key}.\n"
    f"Because this device is tracked by Amazon Cognito, you won't have to re-
enter an MFA code."
)
q.ask("Press Enter when ready.")
auth_tokens = cog_wrapper.sign_in_with_tracked_device(
    user_name, password, device_key, device_group_key, device_password,
aws_srp
)
print("You're signed in. Your access token is:")
pp(auth_tokens["AccessToken"])
print("-" * 88)

print("Don't forget to delete your user pool when you're done with this
example.")
print("\nThanks for watching!")
print("-" * 88)

def main():
    parser = argparse.ArgumentParser(
        description="Shows how to sign up a new user with Amazon Cognito and
associate "
        "the user with an MFA application for multi-factor authentication."
    )
    parser.add_argument(
        "user_pool_id", help="The ID of the user pool to use for the example."
    )
    parser.add_argument(
        "client_id", help="The ID of the client application to use for the
example."
    )
    args = parser.parse_args()
    try:
        run_scenario(boto3.client("cognito-idp"), args.user_pool_id,
args.client_id)
    except Exception:
        logging.exception("Something went wrong with the demo.")

if __name__ == "__main__":
    main()
```

- Untuk detail API, lihat topik berikut ini adalah Referensi API SDK untuk Python (Boto3)AWS
  - [AdminGetUser](#)
  - [AdminInitiateAuth](#)
  - [AdminRespondToAuthChallenge](#)
  - [AssociateSoftwareToken](#)
  - [ConfirmDevice](#)
  - [ConfirmSignUp](#)
  - [InitiateAuth](#)
  - [ListUsers](#)
  - [ResendConfirmationCode](#)
  - [RespondToAuthChallenge](#)
  - [SignUp](#)
  - [VerifySoftwareToken](#)

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.


## Menulis data aktivitas kustom dengan fungsi Lambda setelah autentikasi pengguna Amazon Cognito menggunakan SDK AWS

Contoh kode berikut menunjukkan cara menulis data aktivitas kustom dengan fungsi Lambda setelah autentikasi pengguna Amazon Cognito.

- Gunakan fungsi administrator untuk menambahkan pengguna ke kumpulan pengguna.
- Konfigurasi kumpulan pengguna untuk memanggil fungsi Lambda untuk PostAuthentication pemicunya.
- Masuk pengguna baru ke Amazon Cognito.
- Fungsi Lambda menulis informasi kustom ke CloudWatch Log dan ke tabel DynamoDB.
- Dapatkan dan tampilkan data kustom dari tabel DynamoDB, lalu bersihkan sumber daya.

## Go

## SDK untuk Go V2

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkap dan pelajari cara menyiapkan dan menjalankan di [Repositori Contoh Kode AWS](#).

Jalankan skenario interaktif di penggugah/prompt perintah.

```
// ActivityLog separates the steps of this scenario into individual functions so
// that
// they are simpler to read and understand.
type ActivityLog struct {
    helper      IScenarioHelper
    questioner  demotools.IQuestioner
    resources   Resources
    cognitoActor *actions.CognitoActions
}

// NewActivityLog constructs a new activity log runner.
func NewActivityLog(sdkConfig aws.Config, questioner demotools.IQuestioner,
    helper IScenarioHelper) ActivityLog {
    scenario := ActivityLog{
        helper:      helper,
        questioner:  questioner,
        resources:   Resources{},
        cognitoActor: &actions.CognitoActions{CognitoClient:
cognitoidentityprovider.NewFromConfig(sdkConfig)},
    }
    scenario.resources.init(scenario.cognitoActor, questioner)
    return scenario
}

// AddUserToPool selects a user from the known users table and uses administrator
// credentials to add the user to the user pool.
func (runner *ActivityLog) AddUserToPool(userPoolId string, tableName string)
(string, string) {
    log.Println("To facilitate this example, let's add a user to the user pool using
administrator privileges.")
```

```
users, err := runner.helper.GetKnownUsers(tableName)
if err != nil {
    panic(err)
}
user := users.Users[0]
log.Printf("Adding known user %v to the user pool.\n", user.UserName)
err = runner.cognitoActor.AdminCreateUser(userPoolId, user.UserName,
user.UserEmail)
if err != nil {
    panic(err)
}
pwSet := false
password := runner.questioner.AskPassword("\nEnter a password that has at least
eight characters, uppercase, lowercase, numbers and symbols.\n"+
"(the password will not display as you type):", 8)
for !pwSet {
    log.Printf("\nSetting password for user '%v'.\n", user.UserName)
    err = runner.cognitoActor.AdminSetUserPassword(userPoolId, user.UserName,
password)
    if err != nil {
        var invalidPassword *types.InvalidPasswordException
        if errors.As(err, &invalidPassword) {
            password = runner.questioner.AskPassword("\nEnter another password:", 8)
        } else {
            panic(err)
        }
    } else {
        pwSet = true
    }
}

log.Println(strings.Repeat("-", 88))

return user.UserName, password
}

// AddActivityLogTrigger adds a Lambda handler as an invocation target for the
PostAuthentication trigger.
func (runner *ActivityLog) AddActivityLogTrigger(userPoolId string,
activityLogArn string) {
    log.Println("Let's add a Lambda function to handle the PostAuthentication
trigger from Cognito.\n" +
"This trigger happens after a user is authenticated, and lets your function
take action, such as logging\n" +
```

```
"the outcome.")
err := runner.cognitoActor.UpdateTriggers(
    userPoolId,
    actions.TriggerInfo{Trigger: actions.PostAuthentication, HandlerArn:
aws.String(activityLogArn)})
if err != nil {
    panic(err)
}
runner.resources.triggers = append(runner.resources.triggers,
actions.PostAuthentication)
log.Printf("Lambda function %v added to user pool %v to handle
PostAuthentication Cognito trigger.\n",
    activityLogArn, userPoolId)

log.Println(strings.Repeat("-", 88))
}

// SignInUser signs in as the specified user.
func (runner *ActivityLog) SignInUser(clientId string, userName string, password
string) {
    log.Printf("Now we'll sign in user %v and check the results in the logs and the
DynamoDB table.", userName)
    runner.questioner.Ask("Press Enter when you're ready.")
    authResult, err := runner.cognitoActor.SignIn(clientId, userName, password)
    if err != nil {
        panic(err)
    }
    log.Println("Sign in successful.",
        "The PostAuthentication Lambda handler writes custom information to CloudWatch
Logs.")

    runner.resources.userAccessTokens = append(runner.resources.userAccessTokens,
        *authResult.AccessToken)
}

// GetKnownUserLastLogin gets the login info for a user from the Amazon DynamoDB
table and displays it.
func (runner *ActivityLog) GetKnownUserLastLogin(tableName string, userName
string) {
    log.Println("The PostAuthentication handler also writes login data to the
DynamoDB table.")
    runner.questioner.Ask("Press Enter when you're ready to continue.")
    users, err := runner.helper.GetKnownUsers(tableName)
    if err != nil {
```



```
panic(err)
}
for _, user := range users.Users {
    if user.UserName == userName {
        log.Println("The last login info for the user in the known users table is:")
        log.Printf("\t%+v", *user.LastLogin)
    }
}
log.Println(strings.Repeat("-", 88))
}

// Run runs the scenario.
func (runner *ActivityLog) Run(stackName string) {
    defer func() {
        if r := recover(); r != nil {
            log.Println("Something went wrong with the demo.")
            runner.resources.Cleanup()
        }
    }()

    log.Println(strings.Repeat("-", 88))
    log.Printf("Welcome\n")

    log.Println(strings.Repeat("-", 88))

    stackOutputs, err := runner.helper.GetStackOutputs(stackName)
    if err != nil {
        panic(err)
    }
    runner.resources.userPoolId = stackOutputs["UserPoolId"]
    runner.helper.PopulateUserTable(stackOutputs["TableName"])
    userName, password := runner.AddUserToPool(stackOutputs["UserPoolId"],
        stackOutputs["TableName"])

    runner.AddActivityLogTrigger(stackOutputs["UserPoolId"],
        stackOutputs["ActivityLogFunctionArn"])
    runner.SignInUser(stackOutputs["UserPoolClientId"], userName, password)
    runner.helper.ListRecentLogEvents(stackOutputs["ActivityLogFunction"])
    runner.GetKnownUserLastLogin(stackOutputs["TableName"], userName)

    runner.resources.Cleanup()

    log.Println(strings.Repeat("-", 88))
    log.Println("Thanks for watching!")
}
```

```
log.Println(strings.Repeat("-", 88))
}
```

Tangani PostAuthentication pelatuk dengan fungsi Lambda.

```
const TABLE_NAME = "TABLE_NAME"

// LoginInfo defines structured login data that can be marshalled to a DynamoDB
// format.
type LoginInfo struct {
    UserPoolId string `dynamodbav:"UserPoolId"`
    ClientId   string `dynamodbav:"ClientId"`
    Time      string `dynamodbav:"Time"`
}

// UserInfo defines structured user data that can be marshalled to a DynamoDB
// format.
type UserInfo struct {
    UserName   string `dynamodbav:"UserName"`
    UserEmail  string `dynamodbav:"UserEmail"`
    LastLogin  LoginInfo `dynamodbav:"LastLogin"`
}

// GetKey marshals the user email value to a DynamoDB key format.
func (user UserInfo) GetKey() map[string]dynamodbtypes.AttributeValue {
    userEmail, err := attributevalue.Marshal(user.UserEmail)
    if err != nil {
        panic(err)
    }
    return map[string]dynamodbtypes.AttributeValue{"UserEmail": userEmail}
}

type handler struct {
    dynamoClient *dynamodb.Client
}

// HandleRequest handles the PostAuthentication event by writing custom data to
// the logs and
// to an Amazon DynamoDB table.
```

```
func (h *handler) HandleRequest(ctx context.Context,
    event events.CognitoEventUserPoolsPostAuthentication)
    (events.CognitoEventUserPoolsPostAuthentication, error) {
    log.Printf("Received post authentication trigger from %v for user '%v'",
        event.TriggerSource, event.UserName)
    tableName := os.Getenv(TABLE_NAME)
    user := UserInfo{
        UserName: event.UserName,
        UserEmail: event.Request.UserAttributes["email"],
        LastLogin: LoginInfo{
            UserPoolId: event.UserPoolID,
            ClientId: event.CallerContext.ClientID,
            Time: time.Now().Format(time.UnixDate),
        },
    }
    // Write to CloudWatch Logs.
    fmt.Printf("#%v", user)

    // Also write to an external system. This examples uses DynamoDB to demonstrate.
    userMap, err := attributevalue.MarshalMap(user)
    if err != nil {
        log.Printf("Couldn't marshal to DynamoDB map. Here's why: %v\n", err)
    } else if len(userMap) == 0 {
        log.Printf("User info marshaled to an empty map.")
    } else {
        _, err := h.dynamoClient.PutItem(ctx, &dynamodb.PutItemInput{
            Item: userMap,
            TableName: aws.String(tableName),
        })
        if err != nil {
            log.Printf("Couldn't write to DynamoDB. Here's why: %v\n", err)
        } else {
            log.Printf("Wrote user info to DynamoDB table %v.\n", tableName)
        }
    }

    return event, nil
}

func main() {
    sdkConfig, err := config.LoadDefaultConfig(context.TODO())
    if err != nil {
        log.Panicln(err)
    }
}
```

```

h := handler{
    dynamoClient: dynamodb.NewFromConfig(sdkConfig),
}
lambda.Start(h.HandleRequest)
}

```

Buat struct yang melakukan tugas-tugas umum.

```

// IScenarioHelper defines common functions used by the workflows in this
// example.
type IScenarioHelper interface {
    Pause(secs int)
    GetStackOutputs(stackName string) (actions.StackOutputs, error)
    PopulateUserTable(tableName string)
    GetKnownUsers(tableName string) (actions.UserList, error)
    AddKnownUser(tableName string, user actions.User)
    ListRecentLogEvents(functionName string)
}

// ScenarioHelper contains AWS wrapper structs used by the workflows in this
// example.
type ScenarioHelper struct {
    questioner demotools.IQuestioner
    dynamoActor *actions.DynamoActions
    cfnActor *actions.CloudFormationActions
    cwlActor *actions.CloudWatchLogsActions
    isTestRun bool
}

// NewScenarioHelper constructs a new scenario helper.
func NewScenarioHelper(sdkConfig aws.Config, questioner demotools.IQuestioner)
ScenarioHelper {
    scenario := ScenarioHelper{
        questioner: questioner,
        dynamoActor: &actions.DynamoActions{DynamoClient:
dynamodb.NewFromConfig(sdkConfig)},
        cfnActor: &actions.CloudFormationActions{CfnClient:
cloudformation.NewFromConfig(sdkConfig)},
        cwlActor: &actions.CloudWatchLogsActions{CwlClient:
cloudwatchlogs.NewFromConfig(sdkConfig)},
    }
}

```

```
    }
    return scenario
}

// Pause waits for the specified number of seconds.
func (helper ScenarioHelper) Pause(secs int) {
    if !helper.isTestRun {
        time.Sleep(time.Duration(secs) * time.Second)
    }
}

// GetStackOutputs gets the outputs from the specified CloudFormation stack in a
// structured format.
func (helper ScenarioHelper) GetStackOutputs(stackName string)
(actions.StackOutputs, error) {
    return helper.cfnActor.GetOutputs(stackName), nil
}

// PopulateUserTable fills the known user table with example data.
func (helper ScenarioHelper) PopulateUserTable(tableName string) {
    log.Printf("First, let's add some users to the DynamoDB %v table we'll use for
this example.\n", tableName)
    err := helper.dynamoActor.PopulateTable(tableName)
    if err != nil {
        panic(err)
    }
}

// GetKnownUsers gets the users from the known users table in a structured
// format.
func (helper ScenarioHelper) GetKnownUsers(tableName string) (actions.UserList,
error) {
    knownUsers, err := helper.dynamoActor.Scan(tableName)
    if err != nil {
        log.Printf("Couldn't get known users from table %v. Here's why: %v\n",
tableName, err)
    }
    return knownUsers, err
}

// AddKnownUser adds a user to the known users table.
func (helper ScenarioHelper) AddKnownUser(tableName string, user actions.User) {
    log.Printf("Adding user '%v' with email '%v' to the DynamoDB known users
table...\n",
```

```
    user.UserName, user.UserEmail)
    err := helper.dynamoActor.AddUser(tableName, user)
    if err != nil {
        panic(err)
    }
}

// ListRecentLogEvents gets the most recent log stream and events for the
// specified Lambda function and displays them.
func (helper ScenarioHelper) ListRecentLogEvents(functionName string) {
    log.Println("Waiting a few seconds to let Lambda write to CloudWatch Logs...")
    helper.Pause(10)
    log.Println("Okay, let's check the logs to find what's happened recently with
    your Lambda function.")
    logStream, err := helper.cwlActor.GetLatestLogStream(functionName)
    if err != nil {
        panic(err)
    }
    log.Printf("Getting some recent events from log stream %v\n",
    *logStream.LogStreamName)
    events, err := helper.cwlActor.GetLogEvents(functionName,
    *logStream.LogStreamName, 10)
    if err != nil {
        panic(err)
    }
    for _, event := range events {
        log.Printf("\t%v", *event.Message)
    }
    log.Println(strings.Repeat("-", 88))
}
```

Buat struct yang membungkus tindakan Amazon Cognito.

```
type CognitoActions struct {
    CognitoClient *cognitoidentityprovider.Client
}
```

```
// Trigger and TriggerInfo define typed data for updating an Amazon Cognito
trigger.
type Trigger int

const (
    PreSignUp Trigger = iota
    UserMigration
    PostAuthentication
)

type TriggerInfo struct {
    Trigger    Trigger
    HandlerArn *string
}

// UpdateTriggers adds or removes Lambda triggers for a user pool. When a trigger
is specified with a `nil` value,
// it is removed from the user pool.
func (actor CognitoActions) UpdateTriggers(userPoolId string,
triggers ...TriggerInfo) error {
    output, err := actor.CognitoClient.DescribeUserPool(context.TODO(),
&cognitoidentityprovider.DescribeUserPoolInput{
    UserPoolId: aws.String(userPoolId),
})
    if err != nil {
        log.Printf("Couldn't get info about user pool %v. Here's why: %v\n",
userPoolId, err)
        return err
    }
    lambdaConfig := output.UserPool.LambdaConfig
    for _, trigger := range triggers {
        switch trigger.Trigger {
            case PreSignUp:
                lambdaConfig.PreSignUp = trigger.HandlerArn
            case UserMigration:
                lambdaConfig.UserMigration = trigger.HandlerArn
            case PostAuthentication:
                lambdaConfig.PostAuthentication = trigger.HandlerArn
        }
    }
    _, err = actor.CognitoClient.UpdateUserPool(context.TODO(),
&cognitoidentityprovider.UpdateUserPoolInput{
    UserPoolId:    aws.String(userPoolId),
    LambdaConfig: lambdaConfig,
```

```
    })
    if err != nil {
        log.Printf("Couldn't update user pool %v. Here's why: %v\n", userPoolId, err)
    }
    return err
}
```

```
// SignUp signs up a user with Amazon Cognito.
func (actor CognitoActions) SignUp(clientId string, userName string, password
string, userEmail string) (bool, error) {
    confirmed := false
    output, err := actor.CognitoClient.SignUp(context.TODO(),
&cognitoidentityprovider.SignUpInput{
    ClientId: aws.String(clientId),
    Password: aws.String(password),
    Username: aws.String(userName),
    UserAttributes: []types.AttributeType{
        {Name: aws.String("email"), Value: aws.String(userEmail)},
    },
})
    if err != nil {
        var invalidPassword *types.InvalidPasswordException
        if errors.As(err, &invalidPassword) {
            log.Println(*invalidPassword.Message)
        } else {
            log.Printf("Couldn't sign up user %v. Here's why: %v\n", userName, err)
        }
    } else {
        confirmed = output.UserConfirmed
    }
    return confirmed, err
}
```

```
// SignIn signs in a user to Amazon Cognito using a username and password
authentication flow.
func (actor CognitoActions) SignIn(clientId string, userName string, password
string) (*types.AuthenticationResultType, error) {
    var authResult *types.AuthenticationResultType
    output, err := actor.CognitoClient.InitiateAuth(context.TODO(),
&cognitoidentityprovider.InitiateAuthInput{
```



```
AuthFlow:      "USER_PASSWORD_AUTH",
ClientId:      aws.String(clientId),
AuthParameters: map[string]string{"USERNAME": userName, "PASSWORD": password},
})
if err != nil {
    var resetRequired *types.PasswordResetRequiredException
    if errors.As(err, &resetRequired) {
        log.Println(*resetRequired.Message)
    } else {
        log.Printf("Couldn't sign in user %v. Here's why: %v\n", userName, err)
    }
} else {
    authResult = output.AuthenticationResult
}
return authResult, err
}

// ForgotPassword starts a password recovery flow for a user. This flow typically
// sends a confirmation code
// to the user's configured notification destination, such as email.
func (actor CognitoActions) ForgotPassword(clientId string, userName string)
(*types.CodeDeliveryDetailsType, error) {
    output, err := actor.CognitoClient.ForgotPassword(context.TODO(),
&cognitoidentityprovider.ForgotPasswordInput{
    ClientId: aws.String(clientId),
    Username: aws.String(userName),
})
    if err != nil {
        log.Printf("Couldn't start password reset for user '%v'. Here's why: %v\n",
userName, err)
    }
    return output.CodeDeliveryDetails, err
}

// ConfirmForgotPassword confirms a user with a confirmation code and a new
// password.
func (actor CognitoActions) ConfirmForgotPassword(clientId string, code string,
userName string, password string) error {
    _, err := actor.CognitoClient.ConfirmForgotPassword(context.TODO(),
&cognitoidentityprovider.ConfirmForgotPasswordInput{
```

```
    ClientId:      aws.String(clientId),
    ConfirmationCode: aws.String(code),
    Password:      aws.String(password),
    Username:      aws.String(userName),
  })
  if err != nil {
    var invalidPassword *types.InvalidPasswordException
    if errors.As(err, &invalidPassword) {
      log.Println(*invalidPassword.Message)
    } else {
      log.Printf("Couldn't confirm user %v. Here's why: %v", userName, err)
    }
  }
  return err
}

// DeleteUser removes a user from the user pool.
func (actor CognitoActions) DeleteUser(userAccessToken string) error {
  _, err := actor.CognitoClient.DeleteUser(context.TODO(),
    &cognitoidentityprovider.DeleteUserInput{
      AccessToken: aws.String(userAccessToken),
    })
  if err != nil {
    log.Printf("Couldn't delete user. Here's why: %v\n", err)
  }
  return err
}

// AdminCreateUser uses administrator credentials to add a user to a user pool.
// This method leaves the user
// in a state that requires they enter a new password next time they sign in.
func (actor CognitoActions) AdminCreateUser(userPoolId string, userName string,
  userEmail string) error {
  _, err := actor.CognitoClient.AdminCreateUser(context.TODO(),
    &cognitoidentityprovider.AdminCreateUserInput{
      UserPoolId:      aws.String(userPoolId),
      Username:        aws.String(userName),
      MessageAction:   types.MessageActionTypeSuppress,
      UserAttributes: []types.AttributeType{{Name: aws.String("email"), Value:
        aws.String(userEmail)}}},
```

```
    })
    if err != nil {
        var userExists *types.UsernameExistsException
        if errors.As(err, &userExists) {
            log.Printf("User %v already exists in the user pool.", userName)
            err = nil
        } else {
            log.Printf("Couldn't create user %v. Here's why: %v\n", userName, err)
        }
    }
    return err
}

// AdminSetUserPassword uses administrator credentials to set a password for a
// user without requiring a
// temporary password.
func (actor CognitoActions) AdminSetUserPassword(userPoolId string, userName
string, password string) error {
    _, err := actor.CognitoClient.AdminSetUserPassword(context.TODO(),
&cognitoidentityprovider.AdminSetUserPasswordInput{
    Password:    aws.String(password),
    UserPoolId:  aws.String(userPoolId),
    Username:    aws.String(userName),
    Permanent:   true,
})
    if err != nil {
        var invalidPassword *types.InvalidPasswordException
        if errors.As(err, &invalidPassword) {
            log.Println(*invalidPassword.Message)
        } else {
            log.Printf("Couldn't set password for user %v. Here's why: %v\n", userName,
err)
        }
    }
    return err
}
```

Buat struct yang membungkus tindakan DynamoDB.

```
// DynamoActions encapsulates the Amazon Simple Notification Service (Amazon SNS)
actions
// used in the examples.
type DynamoActions struct {
    DynamoClient *dynamodb.Client
}

// User defines structured user data.
type User struct {
    UserName string
    UserEmail string
    LastLogin *LoginInfo `dynamodbav:",omitempty"`
}

// LoginInfo defines structured custom login data.
type LoginInfo struct {
    UserPoolId string
    ClientId string
    Time string
}

// UserList defines a list of users.
type UserList struct {
    Users []User
}

// UserNameList returns the usernames contained in a UserList as a list of
strings.
func (users *UserList) UserNameList() []string {
    names := make([]string, len(users.Users))
    for i := 0; i < len(users.Users); i++ {
        names[i] = users.Users[i].UserName
    }
    return names
}

// PopulateTable adds a set of test users to the table.
func (actor DynamoActions) PopulateTable(tableName string) error {
    var err error
    var item map[string]types.AttributeValue
    var writeReqs []types.WriteRequest
    for i := 1; i < 4; i++ {
```

```

    item, err = attributevalue.MarshalMap(User{UserName: fmt.Sprintf("test_user_
%v", i), UserEmail: fmt.Sprintf("test_email_%v@example.com", i)})
    if err != nil {
        log.Printf("Couldn't marshall user into DynamoDB format. Here's why: %v\n",
err)
        return err
    }
    writeReqs = append(writeReqs, types.WriteRequest{PutRequest:
&types.PutRequest{Item: item}})
}
_, err = actor.DynamoClient.BatchWriteItem(context.TODO(),
&dynamodb.BatchWriteItemInput{
    RequestItems: map[string][]types.WriteRequest{tableName: writeReqs},
})
if err != nil {
    log.Printf("Couldn't populate table %v with users. Here's why: %v\n",
tableName, err)
}
return err
}

// Scan scans the table for all items.
func (actor DynamoActions) Scan(tableName string) (UserList, error) {
    var userList UserList
    output, err := actor.DynamoClient.Scan(context.TODO(), &dynamodb.ScanInput{
        TableName: aws.String(tableName),
    })
    if err != nil {
        log.Printf("Couldn't scan table %v for items. Here's why: %v\n", tableName,
err)
    } else {
        err = attributevalue.UnmarshallListOfMaps(output.Items, &userList.Users)
        if err != nil {
            log.Printf("Couldn't unmarshal items into users. Here's why: %v\n", err)
        }
    }
    return userList, err
}

// AddUser adds a user item to a table.
func (actor DynamoActions) AddUser(tableName string, user User) error {
    userItem, err := attributevalue.MarshalMap(user)
    if err != nil {
        log.Printf("Couldn't marshall user to item. Here's why: %v\n", err)
    }
}

```

```

}
_, err = actor.DynamoClient.PutItem(context.TODO(), &dynamodb.PutItemInput{
    Item:      userItem,
    TableName: aws.String(tableName),
})
if err != nil {
    log.Printf("Couldn't put item in table %v. Here's why: %v", tableName, err)
}
return err
}

```

Buat struct yang membungkus tindakan CloudWatch Log.

```

type CloudWatchLogsActions struct {
    CwlClient *cloudwatchlogs.Client
}

// GetLatestLogStream gets the most recent log stream for a Lambda function.
func (actor CloudWatchLogsActions) GetLatestLogStream(functionName string)
(types.LogStream, error) {
    var logStream types.LogStream
    logGroupName := fmt.Sprintf("/aws/lambda/%s", functionName)
    output, err := actor.CwlClient.DescribeLogStreams(context.TODO(),
    &cloudwatchlogs.DescribeLogStreamsInput{
        Descending:  aws.Bool(true),
        Limit:        aws.Int32(1),
        LogGroupName: aws.String(logGroupName),
        OrderBy:     types.OrderByLastEventTime,
    })
    if err != nil {
        log.Printf("Couldn't get log streams for log group %v. Here's why: %v\n",
        logGroupName, err)
    } else {
        logStream = output.LogStreams[0]
    }
    return logStream, err
}

// GetLogEvents gets the most recent eventCount events from the specified log
stream.

```

```

func (actor CloudWatchLogsActions) GetLogEvents(functionName string,
logStreamName string, eventCount int32) (
[]types.OutputLogEvent, error) {
var events []types.OutputLogEvent
logGroupName := fmt.Sprintf("/aws/lambda/%s", functionName)
output, err := actor.CwlClient.GetLogEvents(context.TODO(),
&cloudwatchlogs.GetLogEventsInput{
LogStreamName: aws.String(logStreamName),
Limit:         aws.Int32(eventCount),
LogGroupName:  aws.String(logGroupName),
})
if err != nil {
log.Printf("Couldn't get log event for log stream %v. Here's why: %v\n",
logStreamName, err)
} else {
events = output.Events
}
return events, err
}

```

Buat struct yang membungkus tindakan AWS CloudFormation .

```

// StackOutputs defines a map of outputs from a specific stack.
type StackOutputs map[string]string

type CloudFormationActions struct {
CfnClient *cloudformation.Client
}

// GetOutputs gets the outputs from a CloudFormation stack and puts them into a
structured format.
func (actor CloudFormationActions) GetOutputs(stackName string) StackOutputs {
output, err := actor.CfnClient.DescribeStacks(context.TODO(),
&cloudformation.DescribeStacksInput{
StackName: aws.String(stackName),
})
if err != nil || len(output.Stacks) == 0 {
log.Panicf("Couldn't find a CloudFormation stack named %v. Here's why: %v\n",
stackName, err)
}
}

```

```

stackOutputs := StackOutputs{}
for _, out := range output.Stacks[0].Outputs {
    stackOutputs[*out.OutputKey] = *out.OutputValue
}
return stackOutputs
}

```

## Pembersihan sumber daya

```

// Resources keeps track of AWS resources created during an example and handles
// cleanup when the example finishes.
type Resources struct {
    userPoolId      string
    userAccessTokens []string
    triggers        []actions.Trigger

    cognitoActor *actions.CognitoActions
    questioner   demotools.IQuestioner
}

func (resources *Resources) init(cognitoActor *actions.CognitoActions, questioner
demotools.IQuestioner) {
    resources.userAccessTokens = []string{}
    resources.triggers = []actions.Trigger{}
    resources.cognitoActor = cognitoActor
    resources.questioner = questioner
}

// Cleanup deletes all AWS resources created during an example.
func (resources *Resources) Cleanup() {
    defer func() {
        if r := recover(); r != nil {
            log.Printf("Something went wrong during cleanup.\n%v\n", r)
            log.Println("Use the AWS Management Console to remove any remaining resources
\n" +
                "that were created for this scenario.")
        }
    }()
}

```



```
wantDelete := resources.questioner.AskBool("Do you want to remove all of the AWS
resources that were created "+
"during this demo (y/n)?", "y")
if wantDelete {
    for _, accessToken := range resources.userAccessTokens {
        err := resources.cognitoActor.DeleteUser(accessToken)
        if err != nil {
            log.Println("Couldn't delete user during cleanup.")
            panic(err)
        }
        log.Println("Deleted user.")
    }
    triggerList := make([]actions.TriggerInfo, len(resources.triggers))
    for i := 0; i < len(resources.triggers); i++ {
        triggerList[i] = actions.TriggerInfo{Trigger: resources.triggers[i],
HandlerArn: nil}
    }
    err := resources.cognitoActor.UpdateTriggers(resources.userPoolId,
triggerList...)
    if err != nil {
        log.Println("Couldn't update Cognito triggers during cleanup.")
        panic(err)
    }
    log.Println("Removed Cognito triggers from user pool.")
} else {
    log.Println("Be sure to remove resources when you're done with them to avoid
unexpected charges!")
}
}
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK for Go .
  - [AdminCreateUser](#)
  - [AdminSetUserPassword](#)
  - [DeleteUser](#)
  - [InitiateAuth](#)
  - [UpdateUserPool](#)

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

## Contoh kode untuk Sinkronisasi Amazon Cognito menggunakan SDK AWS

Contoh kode berikut menunjukkan cara menggunakan Amazon Cognito Sync dengan AWS perangkat pengembangan perangkat lunak (SDK).

Tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Meskipun tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks pada skenario terkait dan contoh lintas layanan.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Contoh kode

- [Tindakan untuk Sinkronisasi Amazon Cognito menggunakan SDK AWS](#)
  - [Gunakan ListIdentityPoolUsage dengan AWS SDK atau CLI](#)

## Tindakan untuk Sinkronisasi Amazon Cognito menggunakan SDK AWS

Contoh kode berikut menunjukkan cara melakukan tindakan Sinkronisasi Amazon Cognito individual dengan AWS SDK. Kutipan ini memanggil Amazon Cognito Sync 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 Sinkronisasi Amazon Cognito](#).

Contoh

- [Gunakan ListIdentityPoolUsage dengan AWS SDK atau CLI](#)

## Gunakan `ListIdentityPoolUsage` dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan `ListIdentityPoolUsage`.

Rust

SDK untuk Rust

### Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [AWS Repositori Contoh Kode](#).

```
async fn show_pools(client: &Client) -> Result<(), Error> {
    let response = client
        .list_identity_pool_usage()
        .max_results(10)
        .send()
        .await?;

    let pools = response.identity_pool_usages();
    println!("Identity pools:");

    for pool in pools {
        println!(
            "  Identity pool ID:    {}",
            pool.identity_pool_id().unwrap_or_default()
        );
        println!(
            "  Data storage:         {}",
            pool.data_storage().unwrap_or_default()
        );
        println!(
            "  Sync sessions count: {}",
            pool.sync_sessions_count().unwrap_or_default()
        );
        println!(
            "  Last modified:       {}",
            pool.last_modified_date().unwrap().to_chrono_utc()?
        );
        println!();
    }
}
```

```
    }  
  
    println!("Next token: {}", response.next_token().unwrap_or_default());  
  
    Ok(())  
}
```

- Untuk detail API, lihat [ListIdentityPoolUsagereferensi AWS SDK for Rust API](#).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan layanan ini dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

# Praktik terbaik aplikasi multi-penyewa

Kumpulan pengguna Amazon Cognito beroperasi dengan aplikasi multi-penyewa yang menghasilkan volume permintaan yang harus tetap berada dalam kuota Amazon Cognito. Untuk meningkatkan kapasitas ini ketika basis pelanggan Anda tumbuh, Anda dapat [membeli kapasitas kuota tambahan](#).

## Note

[Kuota Amazon Cognito diterapkan per](#) dan. Akun AWS Wilayah AWS Kuota ini dibagi di semua penyewa di dalam aplikasi Anda. Tinjau kuota layanan Amazon Cognito, dan pastikan kuota memenuhi volume yang diharapkan dan jumlah penyewa yang diharapkan dalam aplikasi Anda.

Bagian ini menjelaskan metode yang dapat Anda terapkan untuk memisahkan penyewa antara sumber daya Amazon Cognito dalam Wilayah yang sama dan. Akun AWS Anda juga dapat membagi penyewa Anda di lebih dari satu Akun AWS atau Wilayah, dan memberikan masing-masing kuota mereka sendiri. Keuntungan lain dari multi-penyewaan multi-wilayah termasuk tingkat isolasi setinggi mungkin, waktu transit jaringan terpendek untuk pengguna yang didistribusikan secara global, dan kepatuhan terhadap model distribusi yang ada di organisasi Anda.

Single-Region multi-tenancy juga dapat memiliki keuntungan bagi pelanggan dan administrator Anda.

Daftar berikut mencakup beberapa keuntungan dari multi-tenancy dengan sumber daya bersama.

Keuntungan dari multi-tenancy

Direktori pengguna umum

Multi-tenancy mendukung model di mana pelanggan memiliki akun di lebih dari satu aplikasi. Anda dapat [menautkan identitas dari penyedia pihak ketiga](#) ke dalam satu profil kumpulan pengguna yang konsisten. Dalam kasus di mana profil pengguna unik untuk penyewa mereka, setiap strategi multi-tenancy dengan kumpulan pengguna tunggal memiliki satu titik masuk ke administrasi pengguna.

Keamanan umum

Dalam kumpulan pengguna bersama, Anda dapat membuat satu standar untuk keamanan dan menerapkan [keamanan lanjutan](#), [otentikasi multi-faktor](#) (MFA), dan [AWS WAF](#) standar yang sama

untuk semua penyewa. Karena ACL AWS WAF web harus Wilayah AWS sama dengan sumber daya yang Anda kaitkan dengannya, multi-tenancy menawarkan akses bersama ke sumber daya yang kompleks. Jika Anda ingin mempertahankan konfigurasi keamanan yang konsisten di aplikasi Amazon Cognito Multi-wilayah, Anda harus menerapkan standar operasional yang mereplikasi konfigurasi Anda di antara sumber daya.

## Kustomisasi umum

Anda dapat menyesuaikan kumpulan pengguna dan kumpulan identitas dengan AWS Lambda. Konfigurasi [pemicu Lambda](#) di kumpulan pengguna dan [peristiwa Amazon](#) Cognito di kumpulan identitas dapat menjadi kompleks. Fungsi Lambda harus Wilayah AWS sama dengan kumpulan pengguna atau kumpulan identitas Anda. Fungsi Lambda bersama dapat menerapkan standar untuk alur autentikasi kustom, migrasi pengguna, pembuatan token, dan fungsi lainnya dalam Wilayah.

## Pesan umum

Amazon Simple Notification Service (Amazon SNS) memerlukan konfigurasi tambahan di Wilayah sebelum Anda dapat [mengirim pesan SMS ke pengguna](#) Anda. Anda dapat mengirim [pesan email](#) dengan identitas dan domain terverifikasi Amazon Simple Email Service (Amazon SES) yang terverifikasi yang terdapat dalam suatu Wilayah.

Dengan multi-tenancy, Anda dapat membagikan konfigurasi dan overhead pemeliharaan ini di antara semua penyewa Anda. Karena Amazon SNS dan Amazon SES tidak tersedia secara keseluruhan Wilayah AWS, membagi sumber daya Anda antar Wilayah memerlukan pertimbangan tambahan.

Saat Anda menggunakan [penyedia pesan khusus](#), Anda mendapatkan penyesuaian umum dari satu fungsi Lambda untuk mengelola pengiriman pesan Anda.

[UI yang dihosting](#) menetapkan cookie sesi di browser sehingga mengenali pengguna yang telah diautentikasi. Saat Anda mengautentikasi pengguna lokal di kumpulan pengguna, cookie sesi mereka mengautentikasi mereka untuk semua klien aplikasi di kumpulan pengguna yang sama. Pengguna lokal ada secara eksklusif di direktori kumpulan pengguna Anda tanpa federasi melalui iDP eksternal. Cookie sesi berlaku selama satu jam. Anda tidak dapat mengubah durasi cookie sesi.

Ada dua cara untuk mencegah login di seluruh klien aplikasi dengan cookie sesi UI yang dihosting.

- Pisahkan pengguna Anda ke dalam kumpulan pengguna per penyewa.
- Ganti login UI yang dihosting dengan login API kumpulan pengguna Amazon Cognito.

## Topik

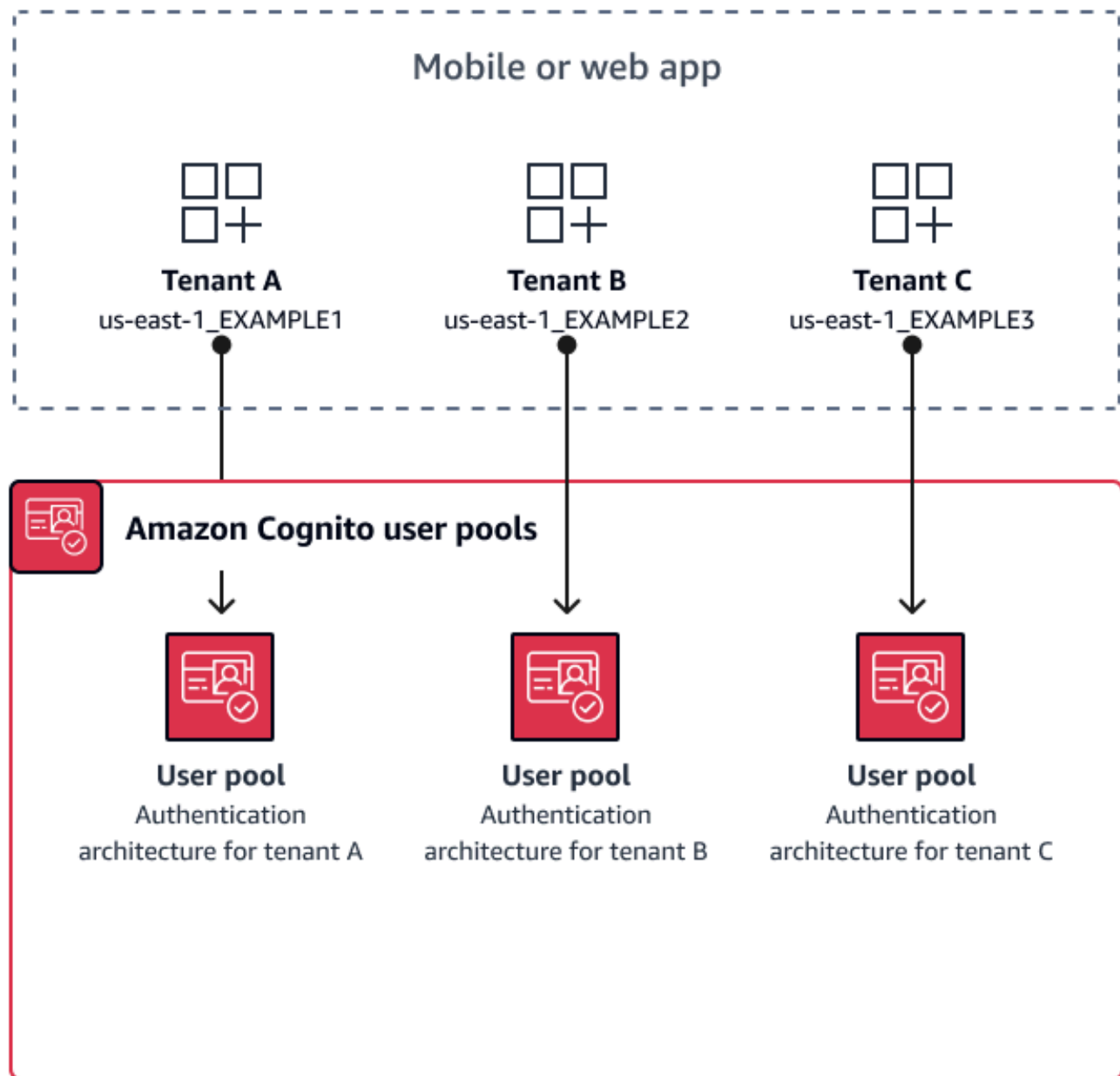
- [Praktik terbaik multi-penyewaan kumpulan pengguna](#)
- [Praktik terbaik multi-tenancy aplikasi-klien](#)
- [Praktik terbaik multi-tenancy grup kumpulan pengguna](#)
- [Praktik terbaik multi-tenancy atribut khusus](#)
- [Rekomendasi keamanan multi-penyewa](#)

## Praktik terbaik multi-penyewaan kumpulan pengguna

Buat kumpulan pengguna untuk setiap penyewa di aplikasi Anda. Pendekatan ini memberikan isolasi maksimum untuk setiap penyewa. Anda dapat menerapkan konfigurasi yang berbeda untuk setiap penyewa. Isolasi penyewa oleh kumpulan pengguna memberi Anda fleksibilitas dalam user-to-tenant pemetaan. Anda dapat membuat beberapa profil untuk pengguna yang sama. Namun, setiap pengguna harus mendaftar secara individual untuk setiap penyewa yang dapat mereka akses.

Dengan menggunakan pendekatan ini, Anda dapat menyiapkan UI yang dihosting untuk setiap penyewa secara independen dan mengarahkan pengguna ke instance khusus penyewa aplikasi Anda. Anda juga dapat menggunakan pendekatan ini untuk berintegrasi dengan layanan backend seperti [Amazon API Gateway](#).

Diagram berikut menunjukkan setiap penyewa dengan kumpulan pengguna khusus.



## Kapan menerapkan multi-tenancy kumpulan pengguna

Ketika isolasi dan kustomisasi menjadi perhatian utama Anda. Hubungan antara pengguna dan penyewa mungkin rumit dalam arsitektur dengan beberapa kumpulan pengguna. Pertimbangkan contoh di mana Anda memiliki dua penyewa pendidikan. Pengguna yang sama mungkin siswa dengan akses terbatas di satu aplikasi, dan guru dengan izin tingkat tinggi di aplikasi lain. Anda mungkin memerlukan MFA di satu aplikasi tetapi tidak yang lain, atau memiliki kebijakan kata



sandi yang berbeda. Karena pengguna lokal dapat masuk ke beberapa klien aplikasi di kumpulan pengguna dengan UI yang dihosting, multi-tenancy kumpulan pengguna juga ideal jika Anda ingin lebih dari satu penyewa masuk dengan UI yang dihosting.

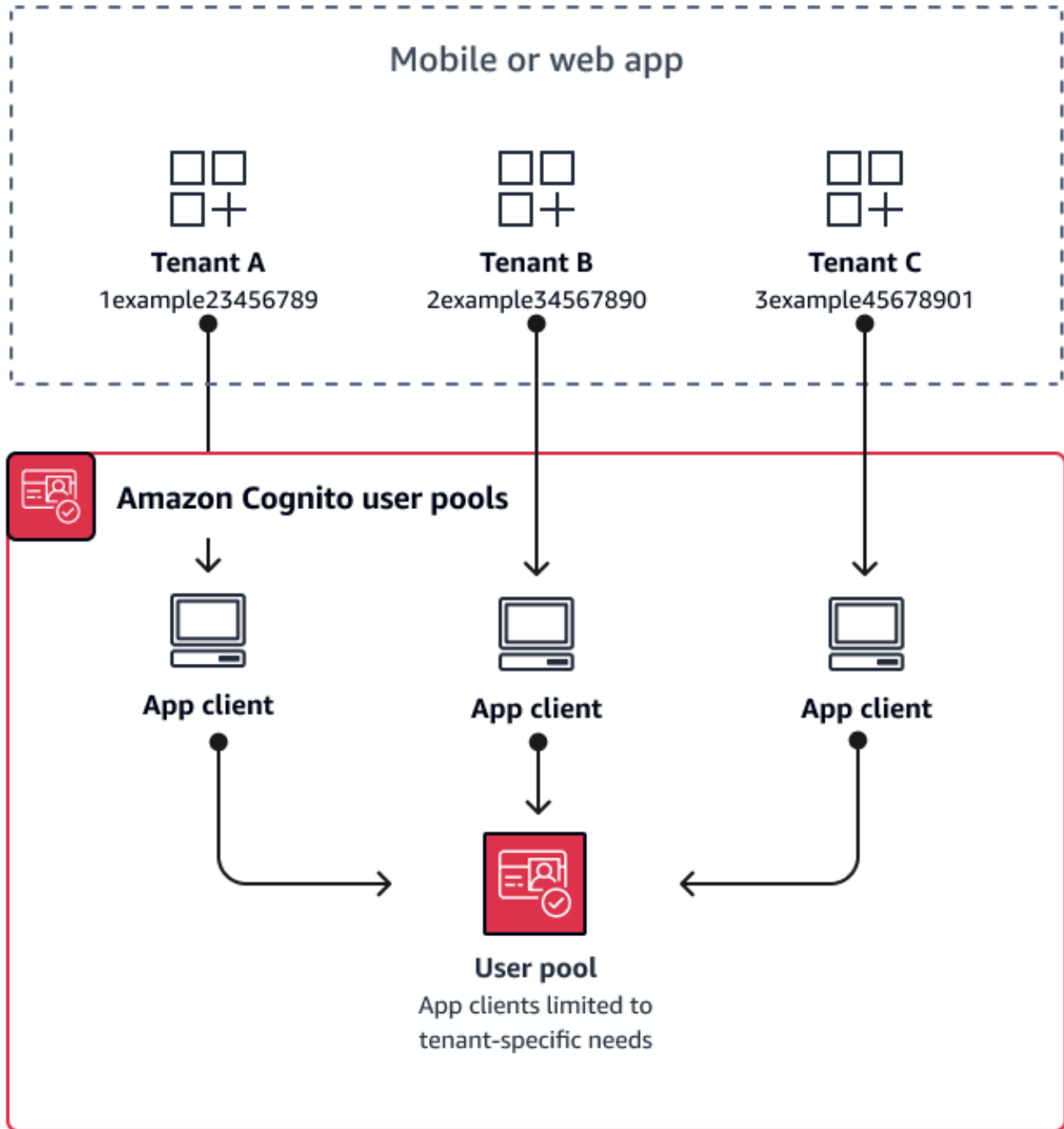
## Tingkat usaha

Upaya pengembangan dan operasi untuk menggunakan pendekatan ini tinggi. Untuk memastikan hasil yang konsisten dan dapat diprediksi untuk keluarga aplikasi Anda, Anda harus mengintegrasikan sumber daya Amazon Cognito dengan alat otomatisasi dan mempertahankan garis dasar Anda saat arsitektur otentikasi Anda semakin kompleks. Saat Anda ingin membuat satu tempat awal untuk aplikasi Anda, Anda harus membangun elemen antarmuka pengguna (UI) untuk menangkap keputusan awal yang mengarahkan pengguna ke sumber daya yang benar.

## Praktik terbaik multi-tenancy aplikasi-klien

Buat [klien aplikasi](#) untuk setiap penyewa di aplikasi Anda. Dengan multi-tenancy app-client, Anda dapat menetapkan pengguna mana pun ke klien aplikasi terkait penyewa dan mempertahankan satu profil pengguna. Karena Anda dapat menetapkan salah satu atau semua [penyedia identitas \(IdPs\)](#) di kumpulan pengguna Anda ke klien aplikasi, klien aplikasi penyewa dapat mengizinkan login dengan iDP khusus penyewa. Ketika pengguna ada di beberapa penyewa, Anda dapat menautkan profil mereka dengan beberapa IdPs untuk pengalaman pengguna yang konsisten.

Diagram berikut menunjukkan setiap penyewa dengan klien aplikasi khusus dalam kumpulan pengguna bersama.



### Kapan menerapkan multi-tenancy aplikasi-klien

Bila Anda dapat memilih konfigurasi universal untuk pengaturan di tingkat kumpulan pengguna, seperti pemicu Lambda, kebijakan kata sandi, dan metode konten dan pengiriman pesan email dan

SMS. Karena pengguna dalam kumpulan pengguna bersama dapat masuk ke klien aplikasi apa pun, multi-tenancy klien aplikasi sangat ideal untuk login dengan atau API kumpulan pengguna app-client-specific IdPs Amazon Cognito. App-client multi-tenancy juga cocok untuk one-to-many lingkungan di mana Anda ingin mengizinkan pengguna untuk transisi antara beberapa aplikasi.

## Tingkat usaha

Multi-tenancy aplikasi-klien membutuhkan upaya moderat. Tantangan utama multi-tenancy aplikasi-klien adalah kemampuan penyewa untuk menyajikan cookie UI yang dihosting dan beralih antar aplikasi. Dalam arsitektur multi-tenancy aplikasi-klien, hindari login UI yang dihosting jika diperlukan isolasi. Anda dapat mendistribusikan aplikasi seluler atau tautan ke aplikasi web Anda dengan logika klien aplikasi bawaan, atau Anda dapat membuat elemen UI awal yang menentukan penyewaan pengguna. Tingkat usaha lebih rendah karena Anda tidak perlu menstandarisasi dan mempertahankan konfigurasi di beberapa kumpulan pengguna dan kumpulan identitas.

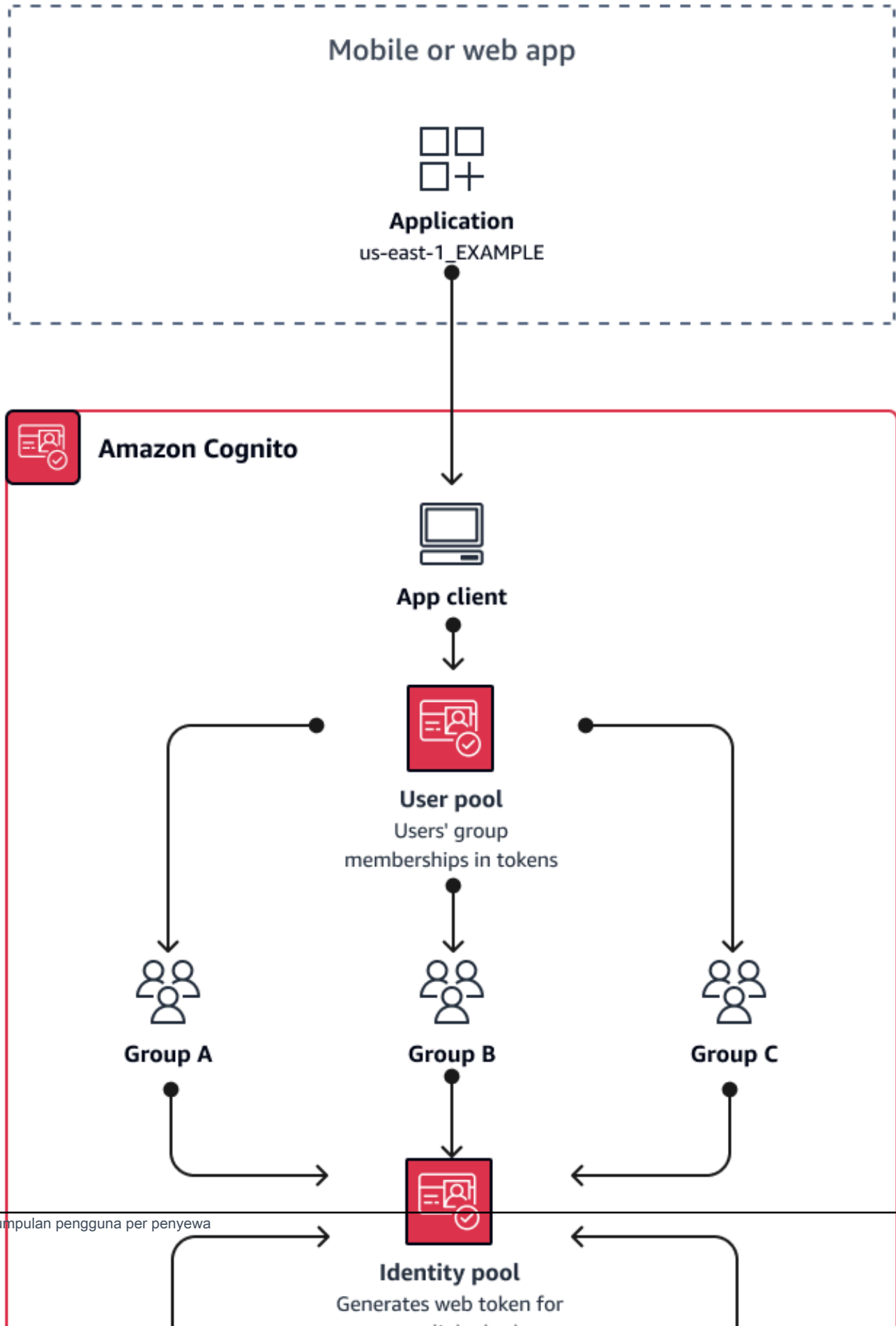
## Praktik terbaik multi-tenancy grup kumpulan pengguna

Multi-tenancy berbasis grup berfungsi paling baik saat arsitektur Anda memerlukan kumpulan pengguna Amazon Cognito dengan kumpulan identitas.

[ID kumpulan pengguna dan token akses](#) berisi `cognito:groups` klaim. Selain itu, token ID berisi `cognito:roles` dan `cognito:preferred_role` klaim. Jika hasil utama autentikasi di aplikasi Anda adalah AWS kredensi sementara dari kumpulan identitas, keanggotaan grup pengguna Anda dapat menentukan [peran IAM](#) dan izin yang mereka terima.

Sebagai contoh, pertimbangkan tiga penyewa yang masing-masing menyimpan aset aplikasi di bucket Amazon S3 mereka sendiri. Tetapkan pengguna setiap penyewa ke grup terkait, konfigurasi peran yang disukai untuk grup, dan berikan akses baca peran tersebut ke bucket mereka.

Diagram berikut menunjukkan penyewa berbagi klien aplikasi dan kumpulan pengguna, dengan grup khusus di kumpulan pengguna yang menentukan kelayakan mereka untuk peran IAM.



## Kapan menerapkan multi-tenancy grup

Ketika akses ke AWS sumber daya adalah perhatian utama Anda. Grup di kumpulan pengguna Amazon Cognito kumpulan pengguna adalah mekanisme untuk kontrol akses berbasis peran (RBAC). Anda dapat mengonfigurasi banyak grup dalam kumpulan pengguna dan membuat keputusan RBAC yang kompleks dengan prioritas grup. Kumpulan identitas dapat menetapkan kredensial untuk peran dengan prioritas tertinggi, peran apa pun dalam klaim grup, atau dari klaim lain dalam token pengguna.

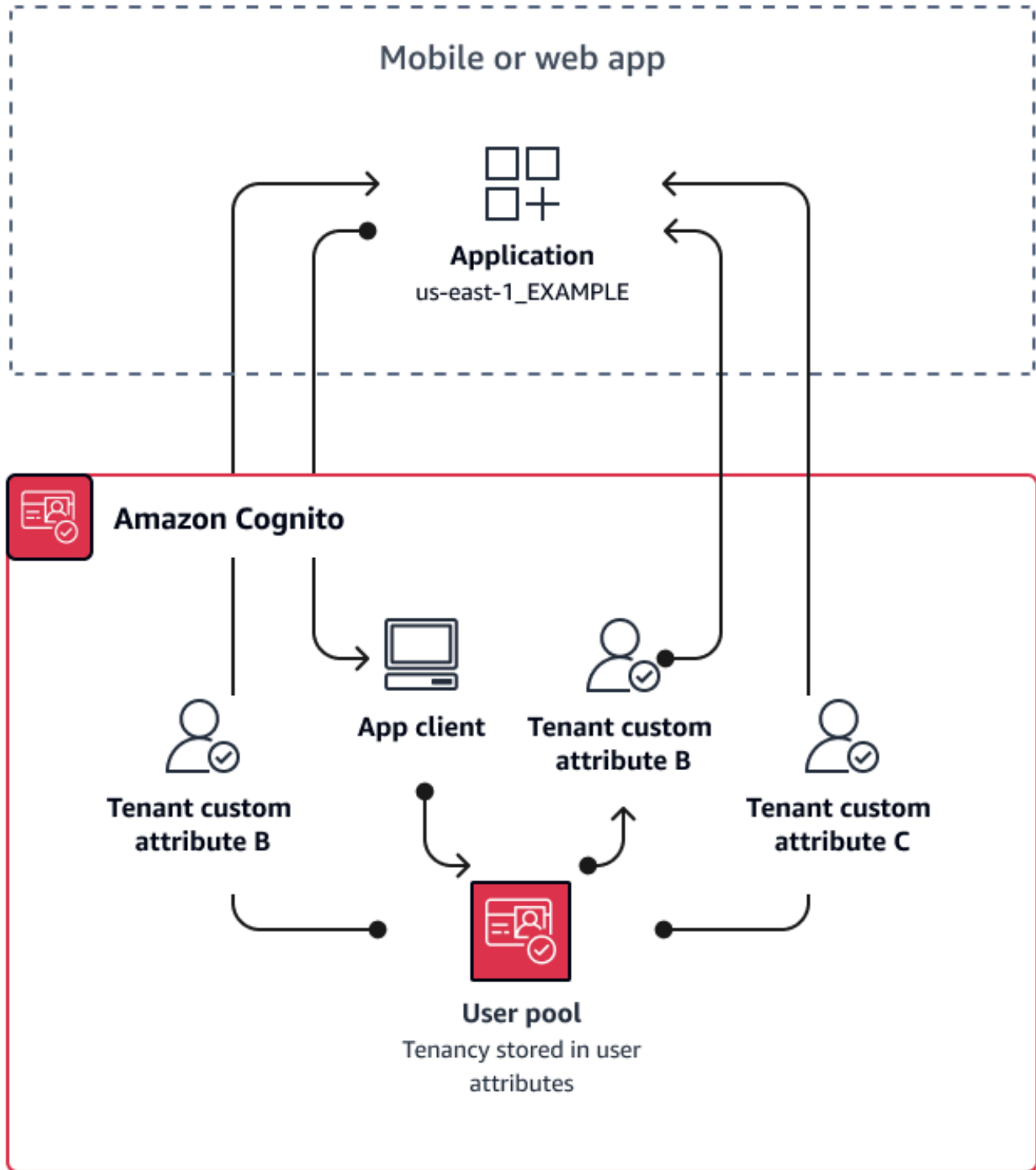
### Tingkat usaha

Tingkat upaya untuk mempertahankan multi-tenancy dengan keanggotaan kelompok saja rendah. Namun, untuk memperluas peran grup kumpulan pengguna di luar kapasitas bawaan untuk pemilihan peran IAM, Anda harus membangun logika aplikasi yang memproses keanggotaan grup dalam token pengguna, dan menentukan apa yang harus dilakukan di klien. Anda dapat mengintegrasikan Izin Terverifikasi Amazon dengan aplikasi Anda untuk membuat keputusan otorisasi sisi klien. Pengidentifikasi grup saat ini tidak diproses dalam operasi [IsAuthorizedWithToken](#) API Izin Terverifikasi, tetapi Anda dapat [mengembangkan kode khusus](#) yang mengurai konten token, termasuk klaim keanggotaan grup.

## Praktik terbaik multi-tenancy atribut khusus

Amazon Cognito mendukung [atribut khusus](#) dengan nama yang Anda pilih. Salah satu skenario di mana atribut kustom berguna adalah ketika mereka membedakan penyewaan pengguna di kumpulan pengguna bersama. Saat Anda menetapkan nilai untuk atribut seperti kepada `penggunacustom:tenantID`, aplikasi Anda dapat menetapkan akses ke sumber daya khusus penyewa. Atribut kustom yang mendefinisikan ID penyewa harus tidak dapat diubah atau hanya-baca untuk klien aplikasi.

Diagram berikut menunjukkan penyewa berbagi klien aplikasi dan kumpulan pengguna, dengan atribut kustom di kumpulan pengguna yang menunjukkan penyewa milik mereka.



Saat atribut kustom menentukan penyewaan, Anda dapat mendistribusikan satu aplikasi atau URL masuk. Setelah pengguna masuk, aplikasi Anda dapat memproses `custom:tenantID` klaim

untuk menentukan aset mana yang akan dimuat, branding yang akan diterapkan, dan fitur yang akan ditampilkan. Untuk keputusan kontrol akses lanjutan dari atribut pengguna, siapkan kumpulan pengguna Anda sebagai penyedia identitas di Izin Terverifikasi Amazon, dan buat keputusan akses dari konten ID atau token akses.

## Kapan menerapkan multi-tenancy atribut khusus

Ketika sewa adalah tingkat permukaan. Atribut penyewa dapat berkontribusi pada hasil branding dan tata letak. Ketika Anda ingin mencapai isolasi yang signifikan antara penyewa, atribut khusus bukanlah pilihan terbaik. Perbedaan apa pun antara penyewa yang harus dikonfigurasi di tingkat kumpulan pengguna atau klien aplikasi, seperti MFA atau pencitraan merek UI yang dihosting, mengharuskan Anda membuat perbedaan antara penyewa dengan cara yang tidak ditawarkan atribut khusus. Dengan kumpulan identitas, Anda bahkan dapat memilih peran IAM dari pengguna Anda dari klaim atribut khusus di token ID mereka.

## Tingkat usaha

Karena multi-tenancy atribut kustom mengalihkan tugas keputusan otorisasi berbasis penyewa di aplikasi Anda, tingkat upayanya cenderung tinggi. Jika Anda sudah berpengalaman dalam konfigurasi klien yang mem-parsing klaim OIDC, atau dalam Izin Terverifikasi Amazon, pendekatan ini mungkin memerlukan tingkat upaya terendah.

# Rekomendasi keamanan multi-penyewa

Untuk membantu membuat aplikasi Anda lebih aman, kami merekomendasikan hal berikut:

- Validasi penyewaan di aplikasi Anda dengan Izin Terverifikasi Amazon. Buat kebijakan yang memeriksa kumpulan pengguna, klien aplikasi, grup, atau hak atribut khusus sebelum Anda mengizinkan permintaan pengguna dalam aplikasi Anda. AWS membuat [sumber identitas Izin Terverifikasi](#) dengan mempertimbangkan kumpulan pengguna Amazon Cognito. Izin Terverifikasi memiliki [panduan tambahan untuk manajemen](#) multi-tenancy.
- Gunakan hanya alamat email terverifikasi untuk mengotorisasi akses pengguna ke penyewa berdasarkan kecocokan domain. Jangan percaya alamat email dan nomor telepon kecuali aplikasi Anda memverifikasinya, atau iDP eksternal memberikan bukti verifikasi. Untuk detail lebih lanjut tentang pengaturan izin ini, lihat [Izin dan Cakupan Atribut](#).
- Gunakan atribut kustom yang tidak dapat diubah, atau hanya-baca, untuk atribut profil pengguna yang mengidentifikasi penyewa. Anda hanya dapat menetapkan nilai atribut yang tidak dapat

diubah saat membuat pengguna atau pengguna mendaftar di kumpulan pengguna Anda. Selain itu, berikan akses hanya-baca kepada klien aplikasi ke atribut.

- Gunakan pemetaan 1:1 antara IDP eksternal penyewa dan klien aplikasi untuk mencegah akses penyewa silang yang tidak sah. Pengguna yang telah diautentikasi oleh IDP eksternal, dan yang memiliki cookie sesi Amazon Cognito yang valid, dapat mengakses aplikasi penyewa lain yang mempercayai IDP yang sama.
- Saat Anda menerapkan logika pencocokan penyewa dan otorisasi dalam aplikasi Anda, batasi pengguna sehingga mereka tidak dapat mengubah kriteria yang mengotorisasi akses pengguna ke penyewa. Juga, jika IDP eksternal digunakan untuk federasi, batasi administrator penyedia identitas penyewa sehingga mereka tidak dapat mengubah akses pengguna.



# Skenario Amazon Cognito yang umum

Topik ini menjelaskan enam skenario umum untuk menggunakan Amazon Cognito.

Dua komponen utama Amazon Cognito adalah kumpulan pengguna dan kumpulan identitas. Kolam pengguna adalah direktori pengguna yang menyediakan opsi pendaftaran dan masuk bagi pengguna aplikasi web dan seluler Anda. Identity pool menyediakan AWS kredensial sementara untuk memberikan pengguna Anda akses ke yang lain. Layanan AWS

Kolam pengguna adalah direktori pengguna di Amazon Cognito. Pengguna aplikasi Anda dapat masuk langsung melalui kumpulan pengguna, atau mereka dapat melakukan federasi melalui penyedia identitas pihak ketiga (iDP). Kumpulan pengguna mengelola overhead penanganan token yang dikembalikan dari login sosial melalui Facebook, Google, Amazon, dan Apple, dan dari OpenID Connect (OIDC) dan SAMP. IdPs Apakah pengguna Anda masuk secara langsung atau melalui pihak ketiga, semua anggota kolam pengguna memiliki profil direktori yang dapat Anda akses melalui SDK.

Dengan kumpulan identitas, pengguna Anda dapat memperoleh AWS kredensial sementara untuk mengakses AWS layanan, seperti Amazon S3 dan DynamoDB. Identity pool mendukung pengguna tamu anonim, serta federasi melalui pihak ketiga IdPs.

Topik

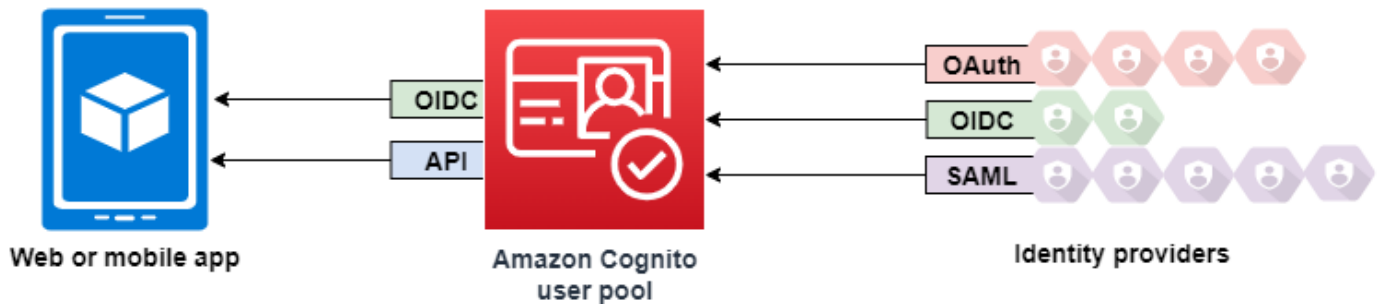
- [Otentikasi dengan kumpulan pengguna](#)
- [Akses sumber daya sisi server Anda dengan kumpulan pengguna](#)
- [Akses sumber daya dengan API Gateway dan Lambda dengan kumpulan pengguna](#)
- [Akses AWS layanan dengan kumpulan pengguna dan kolam identitas](#)
- [Mengautentikasi dengan pihak ketiga dan mengakses AWS layanan dengan kumpulan identitas](#)
- [Akses AWS AppSync sumber daya dengan Amazon Cognito](#)

## Otentikasi dengan kumpulan pengguna

Anda dapat mengaktifkan pengguna untuk melakukan autentikasi dengan kolam pengguna. Pengguna aplikasi Anda dapat masuk langsung melalui kumpulan pengguna, atau mereka dapat melakukan federasi melalui penyedia identitas pihak ketiga (iDP). Kumpulan pengguna mengelola overhead penanganan token yang dikembalikan dari login sosial melalui Facebook, Google, Amazon, dan Apple, dan dari OpenID Connect (OIDC) dan SAMP. IdPs

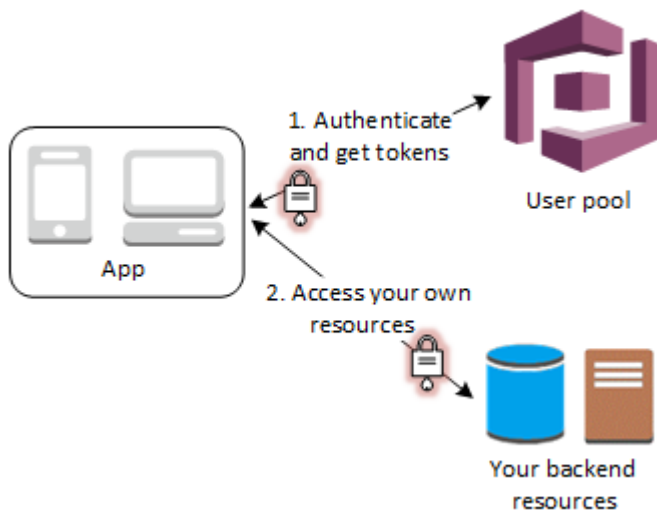
Setelah autentikasi berhasil, aplikasi web atau seluler Anda akan menerima token kolam pengguna dari Amazon Cognito. Anda dapat menggunakan token tersebut untuk mengambil AWS kredensial yang memungkinkan aplikasi mengakses AWS layanan lain, atau Anda dapat memilih untuk menggunakannya untuk mengontrol akses ke sumber daya sisi server, atau ke Amazon API Gateway.

Untuk informasi lebih lanjut, lihat [Alur otentikasi kumpulan pengguna](#) dan [Menggunakan token dengan kolam pengguna](#).



## Akses sumber daya sisi server Anda dengan kumpulan pengguna

Setelah berhasil masuk ke kolam pengguna, aplikasi web atau seluler Anda akan menerima token kolam pengguna dari Amazon Cognito. Anda dapat menggunakan token tersebut untuk mengontrol akses ke sumber daya sisi server Anda. Anda juga dapat membuat grup kolam pengguna untuk mengelola izin, dan untuk mewakili berbagai jenis pengguna. Untuk informasi selengkapnya tentang penggunaan grup untuk mengontrol akses ke sumber daya Anda, lihat [Menambahkan grup ke kumpulan pengguna](#).



Setelah mengonfigurasi domain untuk kumpulan pengguna, Amazon Cognito menyediakan UI web yang dihosting yang memungkinkan Anda menambahkan halaman pendaftaran dan login ke aplikasi. Dengan menggunakan foundation OAuth 2.0 ini, Anda dapat membuat server sumber daya Anda sendiri untuk memungkinkan pengguna mengakses sumber daya yang dilindungi. Untuk informasi selengkapnya, lihat [Cakupan, M2M, dan otorisasi API dengan server sumber daya](#).

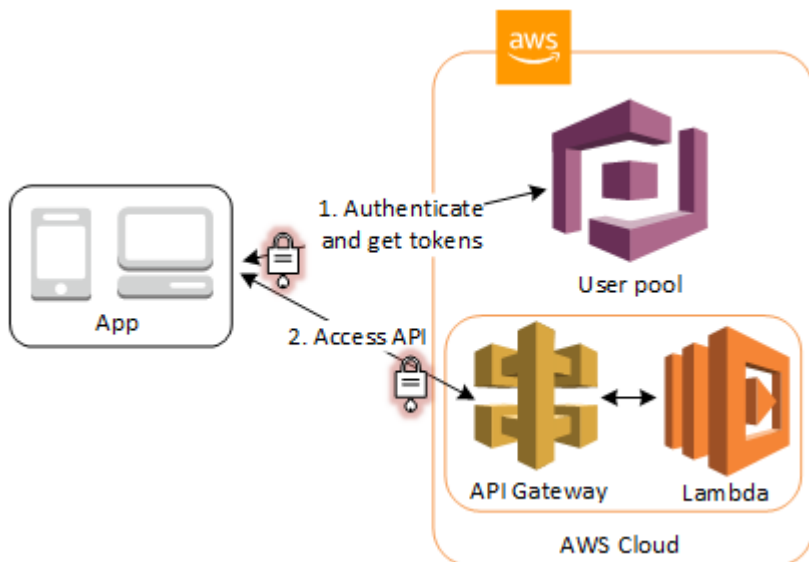
Untuk informasi selengkapnya tentang autentikasi kumpulan pengguna, lihat [Alur otentikasi kumpulan pengguna](#) dan [Menggunakan token dengan kolam pengguna](#).

## Akses sumber daya dengan API Gateway dan Lambda dengan kumpulan pengguna

Anda dapat mengaktifkan pengguna Anda untuk mengakses API Anda melalui API Gateway. API Gateway memvalidasi token dari autentikasi kolam pengguna yang sukses, dan menggunakannya untuk memberi pengguna Anda akses ke sumber daya termasuk fungsi Lambda, atau API Anda sendiri.

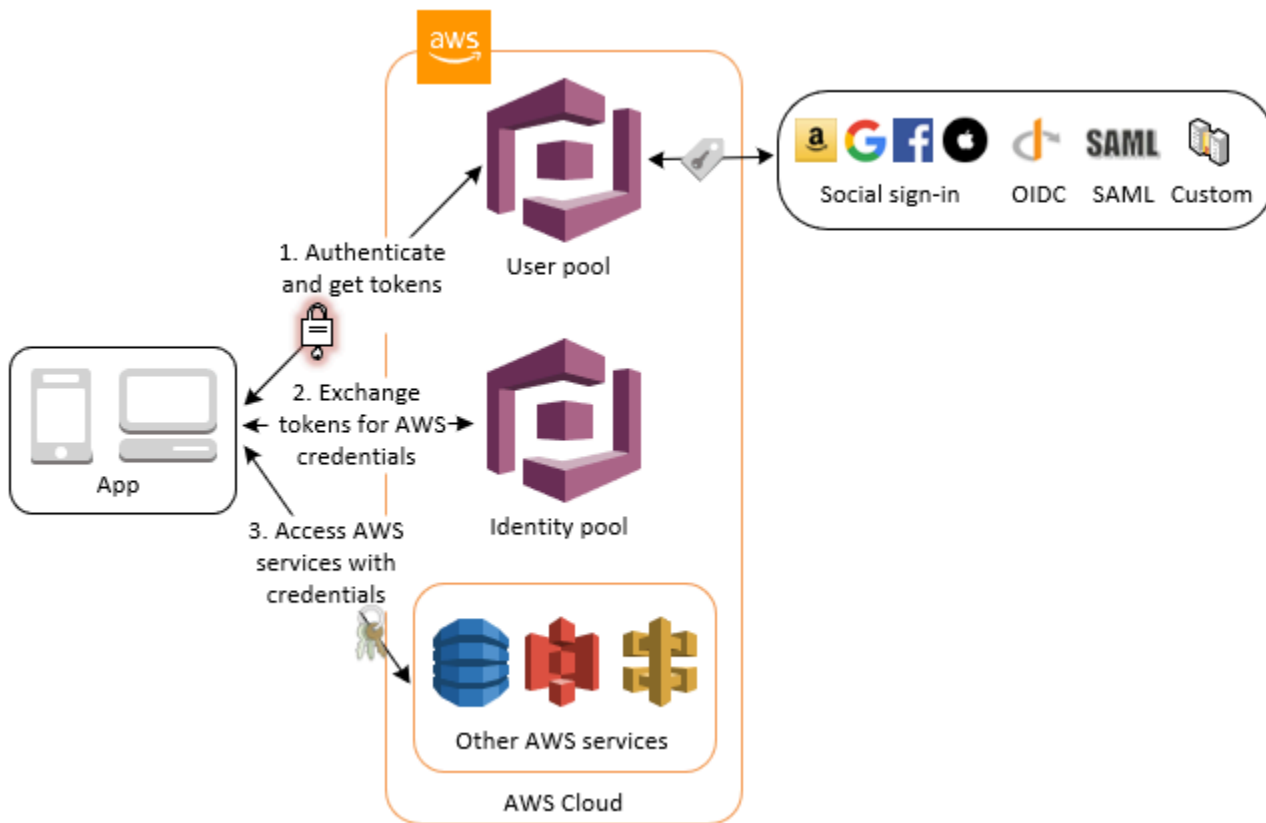
Anda dapat menggunakan grup di kolam pengguna untuk mengontrol izin dengan API Gateway dengan memetakan keanggotaan grup untuk IAM role. Grup yang merupakan anggota pengguna disertakan dalam token ID yang disediakan oleh kolam pengguna saat pengguna aplikasi Anda masuk. Untuk informasi selengkapnya tentang Grup kolam pengguna, lihat [Menambahkan grup ke kumpulan pengguna](#).

Anda dapat mengirimkan token kolam pengguna Anda dengan permintaan ke API Gateway untuk verifikasi oleh fungsi Lambda otorisasi Amazon Cognito. Untuk informasi selengkapnya tentang API Gateway, lihat [Menggunakan API Gateway dengan kolam pengguna Amazon Cognito](#).



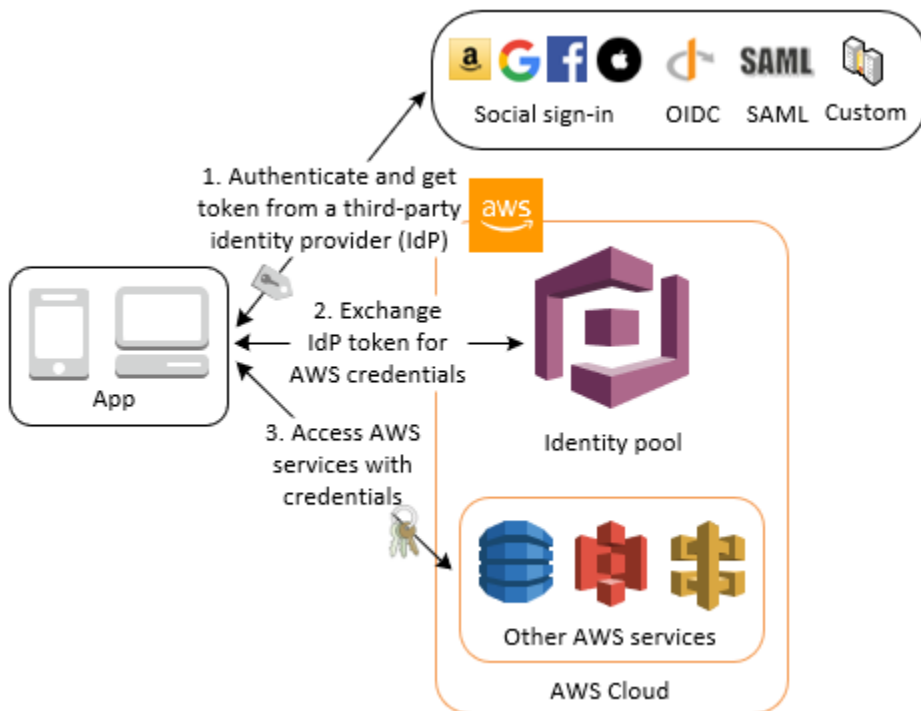
## Akses AWS layanan dengan kumpulan pengguna dan kolom identitas

Setelah autentikasi kolom pengguna yang sukses, aplikasi Anda akan menerima token kolom pengguna dari Amazon Cognito. Anda dapat menukarnya dengan akses sementara ke AWS layanan lain dengan kumpulan identitas. Untuk informasi selengkapnya, lihat [Mengakses Layanan AWS menggunakan kumpulan identitas setelah masuk](#) dan [Memulai dengan kumpulan identitas Amazon Cognito](#).



## Mengautentikasi dengan pihak ketiga dan mengakses AWS layanan dengan kumpulan identitas

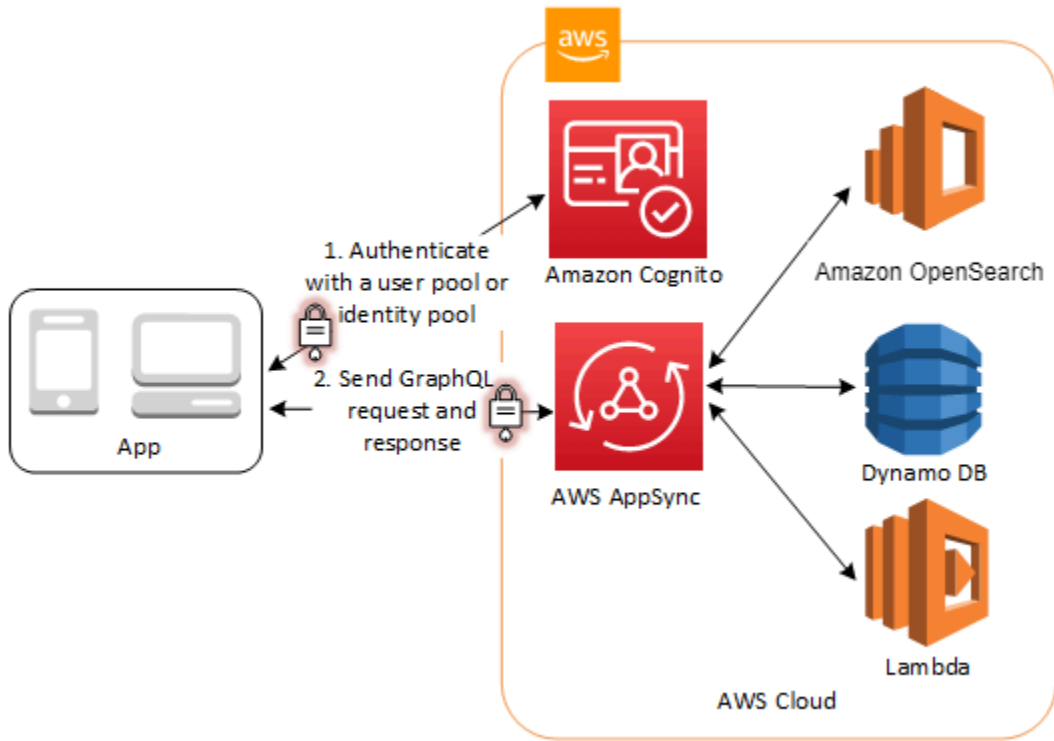
Anda dapat mengaktifkan akses pengguna ke AWS layanan melalui kumpulan identitas. Kumpulan identitas memerlukan token IdP dari pengguna yang diautentikasi oleh penyedia identitas pihak ketiga (atau tidak ada jika tamu anonim). Sebagai gantinya, kumpulan identitas memberikan AWS kredensial sementara yang dapat Anda gunakan untuk mengakses layanan lain. AWS Untuk informasi selengkapnya, lihat [Memulai dengan kumpulan identitas Amazon Cognito](#).



## Akses AWS AppSync sumber daya dengan Amazon Cognito

Anda dapat memberi pengguna akses ke AWS AppSync sumber daya dengan token dari autentikasi kumpulan pengguna Amazon Cognito yang berhasil. Untuk informasi selengkapnya, lihat otorisasi [AMAZON\\_COGNITO\\_USER\\_POOLS](#) di Panduan Pengembang AWS AppSync

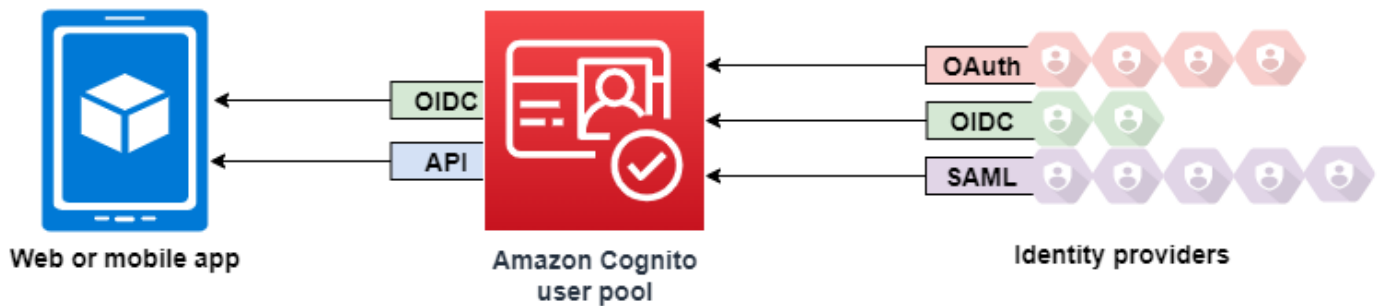
Anda juga dapat menandatangani permintaan ke AWS AppSync GraphQL API dengan kredensial IAM yang Anda terima dari kumpulan identitas. Lihat otorisasi [AWS\\_IAM](#).



# Kolam pengguna Amazon Cognito

Kumpulan pengguna Amazon Cognito adalah direktori pengguna untuk autentikasi dan otorisasi aplikasi web dan seluler. Dari perspektif aplikasi Anda, kumpulan pengguna Amazon Cognito adalah penyedia identitas OpenID Connect (OIDC) (iDP). Kumpulan pengguna menambahkan lapisan fitur tambahan untuk keamanan, federasi identitas, integrasi aplikasi, dan penyesuaian pengalaman pengguna.

Anda dapat, misalnya, memverifikasi bahwa sesi pengguna Anda berasal dari sumber tepercaya. Anda dapat menggabungkan direktori Amazon Cognito dengan penyedia identitas eksternal. Dengan AWS SDK pilihan Anda, Anda dapat memilih model otorisasi API yang paling sesuai untuk aplikasi Anda. Dan Anda dapat menambahkan AWS Lambda fungsi yang memodifikasi atau merombak perilaku default Amazon Cognito.



## Topik

- [Fitur](#)
- [Otentikasi dengan kumpulan pengguna](#)
- [Menggunakan API kumpulan pengguna Amazon Cognito dan titik akhir kumpulan pengguna](#)
- [Memperbarui konfigurasi kumpulan pengguna](#)
- [Menyiapkan dan menggunakan UI yang dihosting Amazon Cognito dan titik akhir federasi](#)
- [Cakupan, M2M, dan otorisasi API dengan server sumber daya](#)
- [Menambahkan login kumpulan pengguna melalui pihak ketiga](#)
- [Menyesuaikan alur kerja kumpulan pengguna dengan pemicu Lambda](#)
- [Menggunakan analitik Amazon Pinpoint dengan kumpulan pengguna Amazon Cognito](#)
- [Mengelola pengguna di kumpulan pengguna Anda](#)
- [Pengaturan email untuk kumpulan pengguna Amazon Cognito](#)



- [Pengaturan pesan SMS untuk kolam pengguna Amazon Cognito](#)
- [Menggunakan token dengan kolam pengguna](#)
- [Mengakses sumber daya setelah otentikasi kumpulan pengguna berhasil](#)
- [Menggunakan fitur keamanan kumpulan pengguna Amazon Cognito](#)

## Fitur

Kumpulan pengguna Amazon Cognito memiliki fitur-fitur berikut.

## Daftar

Kumpulan pengguna Amazon Cognito memiliki metode yang digerakkan oleh pengguna, berbasis administrator, dan terprogram untuk menambahkan profil pengguna ke kumpulan pengguna Anda. Kumpulan pengguna Amazon Cognito mendukung model pendaftaran berikut. Anda dapat menggunakan kombinasi model ini di aplikasi Anda.

### Important

Jika Anda mengaktifkan pendaftaran pengguna di kumpulan pengguna Anda, siapa pun di internet dapat mendaftar untuk akun dan masuk ke aplikasi Anda. Jangan aktifkan registrasi mandiri di kumpulan pengguna kecuali Anda ingin membuka aplikasi untuk pendaftaran publik. Untuk mengubah setelan ini, perbarui Pendaftaran layanan mandiri di tab Pengalaman pendaftaran konsol kumpulan pengguna, atau perbarui nilai permintaan [AllowAdminCreateUserOnly CreateUserPool](#) atau [UpdateUserPool](#) API.

Untuk informasi tentang fitur keamanan yang dapat Anda atur di kumpulan pengguna, lihat [Menggunakan fitur keamanan kumpulan pengguna Amazon Cognito](#).

1. Pengguna Anda dapat memasukkan informasi mereka di aplikasi Anda dan membuat profil pengguna yang asli dari kumpulan pengguna Anda. Anda dapat memanggil operasi pendaftaran API untuk mendaftarkan pengguna di kumpulan pengguna Anda. Anda dapat membuka operasi pendaftaran ini kepada siapa pun, atau Anda dapat mengotorisasi mereka dengan rahasia atau AWS kredensial klien.
2. Anda dapat mengarahkan pengguna ke IDP pihak ketiga yang dapat mereka otorisasi untuk meneruskan informasi mereka ke Amazon Cognito. Amazon Cognito memproses token id OIDC, userInfo data OAuth 2.0, dan pernyataan SAMP 2.0 ke dalam profil pengguna di kumpulan

pengguna Anda. Anda mengontrol atribut yang ingin diterima Amazon Cognito berdasarkan aturan pemetaan atribut.

3. Anda dapat melewati pendaftaran publik atau federasi, dan membuat pengguna berdasarkan sumber data dan skema Anda sendiri. Tambahkan pengguna langsung di konsol Amazon Cognito atau API. Impor pengguna dari file CSV. Jalankan just-in-time AWS Lambda fungsi yang mencari pengguna baru Anda di direktori yang ada, dan mengisi profil pengguna mereka dari data yang ada.

Setelah pengguna mendaftar, Anda dapat menambahkannya ke grup yang dicantumkan Amazon Cognito di token akses dan ID. Anda juga dapat menautkan grup kumpulan pengguna ke peran IAM saat Anda meneruskan token ID ke kumpulan identitas.

Topik terkait

- [Mengelola pengguna di kumpulan pengguna Anda](#)
- [Menggunakan API kumpulan pengguna Amazon Cognito dan titik akhir kumpulan pengguna](#)
- [Contoh kode untuk Penyedia Identitas Amazon Cognito menggunakan SDK AWS](#)

## Masuk

Amazon Cognito dapat menjadi direktori pengguna mandiri dan penyedia identitas (iDP) ke aplikasi Anda. Pengguna Anda dapat masuk dengan UI yang di-host oleh Amazon Cognito, atau dengan UI Anda sendiri melalui API kumpulan pengguna Amazon Cognito. Tingkat aplikasi di belakang UI kustom front end Anda dapat mengotorisasi permintaan di bagian belakang dengan salah satu dari beberapa metode untuk mengonfirmasi permintaan yang sah.

Untuk masuk pengguna dengan direktori eksternal, secara opsional dikombinasikan dengan direktori pengguna bawaan ke Amazon Cognito, Anda dapat menambahkan integrasi berikut.

1. Masuk dan impor data pengguna konsumen dengan login sosial OAuth 2.0. Amazon Cognito mendukung login dengan Google, Facebook, Amazon, dan Apple melalui OAuth 2.0.
2. Masuk dan impor data pengguna perusahaan dengan login SAMP dan OIDC. Anda juga dapat mengonfigurasi Amazon Cognito untuk menerima klaim dari penyedia identitas SAMP atau OpenID Connect (OIDC) (IDP).
3. Tautkan profil pengguna eksternal ke profil pengguna asli. Pengguna yang tertaut dapat masuk dengan identitas pengguna pihak ketiga dan menerima akses yang Anda tetapkan ke pengguna di direktori bawaan.

## Topik terkait

- [Menambahkan login kumpulan pengguna melalui pihak ketiga](#)
- [Menautkan pengguna gabungan ke profil pengguna yang ada](#)

## Machine-to-machine Otorisasi M

Beberapa sesi bukanlah human-to-machine interaksi. Anda mungkin memerlukan akun layanan yang dapat mengotorisasi permintaan ke API dengan proses otomatis. [Untuk menghasilkan token akses untuk machine-to-machine otorisasi dengan cakupan OAuth 2.0, Anda dapat menambahkan klien aplikasi yang menghasilkan hibah kredensial klien.](#)

## Topik terkait

- [Cakupan, M2M, dan otorisasi API dengan server sumber daya](#)

## UI yang Dihosting

Jika Anda tidak ingin membangun antarmuka pengguna, Anda dapat menyajikan UI yang dihosting Amazon Cognito yang disesuaikan kepada pengguna. UI yang dihosting adalah seperangkat halaman web untuk pendaftaran, masuk, otentikasi multi-faktor (MFA), dan pengaturan ulang kata sandi. Anda dapat menambahkan UI yang dihosting ke domain yang ada, atau menggunakan pengenalan awalan di subdomain. AWS

## Topik terkait

- [Menyiapkan dan menggunakan UI yang dihosting Amazon Cognito dan titik akhir federasi](#)
- [Mengkonfigurasi domain kolam pengguna](#)

## Keamanan

Pengguna lokal Anda dapat memberikan faktor otentikasi tambahan dengan kode dari pesan SMS, atau aplikasi yang menghasilkan kode otentikasi multi-faktor (MFA). Anda dapat membangun mekanisme untuk menyiapkan dan memproses MFA di aplikasi Anda, atau Anda dapat membiarkan UI yang dihosting mengelolanya. Kumpulan pengguna Amazon Cognito dapat melewati MFA saat pengguna Anda masuk dari perangkat tepercaya.

Jika Anda tidak ingin awalnya memerlukan MFA dari pengguna Anda, Anda dapat memerlukannya secara kondisional. Dengan fitur keamanan tingkat lanjut, Amazon Cognito dapat mendeteksi potensi aktivitas berbahaya dan mengharuskan pengguna Anda mengatur MFA, atau memblokir proses masuk.

Jika lalu lintas jaringan ke kumpulan pengguna Anda mungkin berbahaya, Anda dapat memantaunya dan mengambil tindakan dengan ACL AWS WAF web.

Topik terkait

- [Menambahkan MFA ke kumpulan pengguna](#)
- [Menambahkan keamanan tingkat lanjut ke kumpulan pengguna](#)
- [Mengaitkan ACL AWS WAF web dengan kumpulan pengguna](#)

## Pengalaman pengguna khusus

Pada sebagian besar tahap pendaftaran, masuk, atau pembaruan profil pengguna, Anda dapat menyesuaikan cara Amazon Cognito menangani permintaan tersebut. Dengan pemicu Lambda, Anda dapat mengubah token ID atau menolak permintaan pendaftaran berdasarkan kondisi khusus. Anda dapat membuat alur otentikasi kustom Anda sendiri.

Anda dapat mengunggah CSS dan logo khusus untuk memberi UI yang dihosting tampilan dan nuansa yang akrab bagi pengguna Anda.

Topik terkait

- [Menyesuaikan alur kerja kumpulan pengguna dengan pemicu Lambda](#)
- [Tantangan otentikasi kustom pemicu Lambda](#)
- [Menyesuaikan halaman web masuk dan daftar bawaan](#)

## Pemantauan dan analitik

Pengguna Amazon Cognito mengumpulkan permintaan API log, termasuk permintaan ke UI yang dihosting, ke. AWS CloudTrail Anda dapat meninjau metrik performa di CloudWatch Log Amazon, mendorong log kustom CloudWatch dengan pemicu Lambda, dan memantau volume permintaan API di konsol Service Quotas.

Anda juga dapat mencatat data perangkat dan sesi dari permintaan API Anda ke kampanye Amazon Pinpoint. Dengan Amazon Pinpoint, Anda dapat mengirim pemberitahuan push dari aplikasi berdasarkan analisis aktivitas pengguna.

Topik terkait

- [Mencatat panggilan API Amazon Cognito dengan AWS CloudTrail](#)
- [Melacak kuota dan penggunaan di CloudWatch dan Service Quotas](#)
- [Menggunakan analitik Amazon Pinpoint dengan kumpulan pengguna Amazon Cognito](#)

## Integrasi kumpulan identitas Amazon Cognito

Setengah lainnya dari Amazon Cognito adalah kumpulan identitas. Kumpulan identitas menyediakan kredensial yang mengotorisasi dan memantau permintaan API Layanan AWS, misalnya Amazon DynamoDB atau Amazon S3, dari pengguna Anda. Anda dapat membuat kebijakan akses berbasis identitas yang melindungi data Anda berdasarkan cara Anda mengklasifikasikan pengguna di kumpulan pengguna Anda. Identity pool juga dapat menerima token dan pernyataan SAMP 2.0 dari berbagai penyedia identitas, terlepas dari otentikasi kumpulan pengguna.

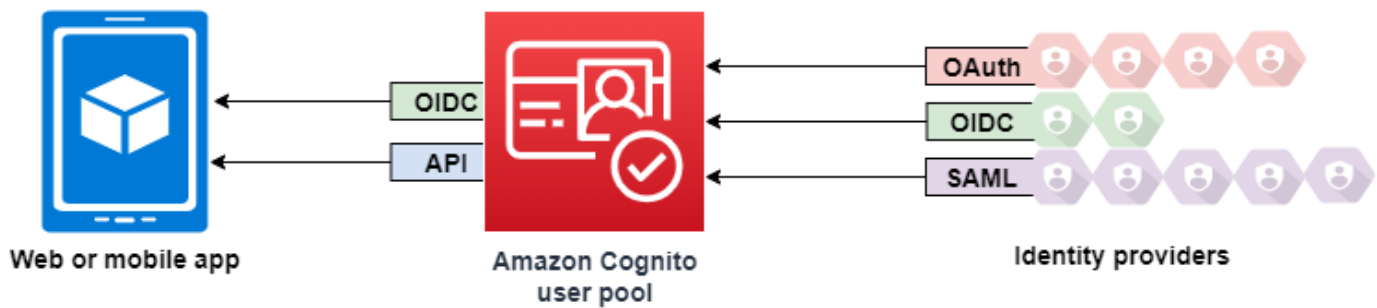
Topik terkait

- [Mengakses Layanan AWS menggunakan kumpulan identitas setelah masuk](#)
- [Kumpulan identitas Amazon Cognito](#)

## Otentikasi dengan kumpulan pengguna

Pengguna aplikasi Anda dapat masuk langsung melalui kumpulan pengguna, atau mereka dapat melakukan federasi melalui penyedia identitas pihak ketiga (iDP). Kumpulan pengguna mengelola overhead penanganan token yang dikembalikan dari login sosial melalui Facebook, Google, Amazon, dan Apple, dan dari OpenID Connect (OIDC) dan SAMP. IdPs

Setelah autentikasi berhasil, Amazon Cognito mengembalikan token kolam pengguna ke aplikasi Anda. Anda dapat menggunakan token untuk memberikan pengguna Anda akses ke sumber daya sisi server Anda sendiri, atau ke Amazon API Gateway. Atau, Anda dapat menukarnya dengan AWS kredensi untuk mengakses layanan lain AWS .



Penanganan dan pengelolaan token kolom pengguna untuk web atau aplikasi seluler Anda disediakan di sisi klien melalui Amazon Cognito SDK. Demikian juga, Mobile SDK for iOS dan Mobile SDK for Android secara otomatis me-refresh ID dan token akses Anda jika ada token refresh yang valid (tidak kedaluwarsa), dan ID serta token akses memiliki sisa validitas minimal 5 menit. Untuk informasi tentang SDK, dan kode contoh untuk, Android JavaScript, dan iOS, lihat SDK kumpulan [pengguna Amazon Cognito](#).

Setelah pengguna aplikasi Anda berhasil masuk, Amazon Cognito membuat sesi dan mengembalikan ID, akses, dan token refresh untuk pengguna yang diautentikasi.

### JavaScript

```
// Amazon Cognito creates a session which includes the id, access, and refresh
tokens of an authenticated user.

var authenticationData = {
    Username : 'username',
    Password : 'password',
};
var authenticationDetails = new
AmazonCognitoIdentity.AuthenticationDetails(authenticationData);
var poolData = { UserPoolId : 'us-east-1_Example',
    ClientId : '1example23456789'
};
var userPool = new AmazonCognitoIdentity.CognitoUserPool(poolData);
var userData = {
    Username : 'username',
    Pool : userPool
};
var cognitoUser = new AmazonCognitoIdentity.CognitoUser(userData);
cognitoUser.authenticateUser(authenticationDetails, {
    onSuccess: function (result) {
```

```

        var accessToken = result.getAccessToken().getJwtToken();

        /* Use the idToken for Logins Map when Federating User Pools with
identity pools or when passing through an Authorization Header to an API Gateway
Authorizer */
        var idToken = result.idToken.jwtToken;
    },

    onFailure: function(err) {
        alert(err);
    },

});

```

## Android

```

// Session is an object of the type CognitoUserSession, and includes the id, access,
and refresh tokens for a user.

String idToken = session.getIdToken().getJWTToken();
String accessToken = session.getAccessToken().getJWT();

```

## iOS - swift

```

// AWSCognitoIdentityUserSession includes id, access, and refresh tokens for a user.

- (AWSTask<AWSCognitoIdentityUserSession *> *)getSession;

```

## iOS - objective-C

```

// AWSCognitoIdentityUserSession includes the id, access, and refresh tokens for a
user.

[[user getSession:@"username" password:@"password" validationData:nil scopes:nil]
continueWithSuccessBlock:^(id _Nullable(AWSTask<AWSCognitoIdentityUserSession *> *
_Nonnull task) {
    // success, task.result has user session
    return nil;
}]];

```

## Topik

- [Alur otentikasi kumpulan pengguna](#)
- [Klien aplikasi kumpulan pengguna](#)
- [Bekerja dengan perangkat pengguna di kumpulan pengguna Anda](#)

## Alur otentikasi kumpulan pengguna

Amazon Cognito mencakup beberapa metode untuk mengautentikasi pengguna Anda. Semua kumpulan pengguna, apakah Anda memiliki domain atau tidak, dapat mengautentikasi pengguna di API kumpulan pengguna. Jika Anda menambahkan domain ke kumpulan pengguna, Anda dapat menggunakan [titik akhir kumpulan pengguna](#). API kumpulan pengguna mendukung berbagai model otorisasi dan alur permintaan untuk permintaan API.

Untuk memverifikasi identitas pengguna, Amazon Cognito mendukung alur otentikasi yang menggabungkan jenis tantangan baru, selain kata sandi. Autentikasi Amazon Cognito biasanya mengharuskan Anda menerapkan dua operasi API dalam urutan berikut:

### Public authentication

1. [InitiateAuth](#)
2. [RespondToAuthChallenge](#)

`InitiateAuth` dan `RespondToAuthChallenge` merupakan API yang tidak diautentikasi untuk digunakan dengan klien aplikasi publik sisi klien.

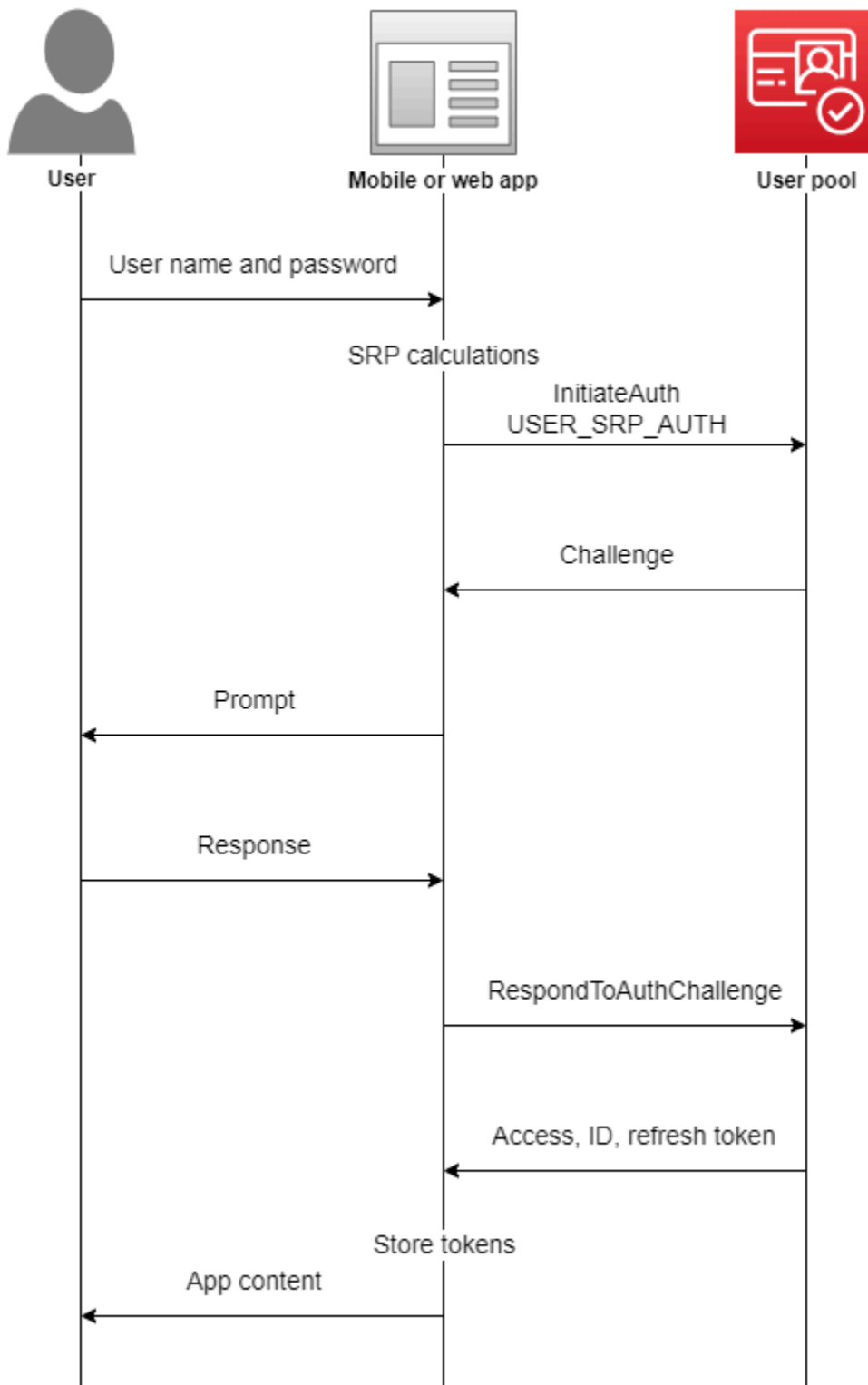
### Server-side authentication

1. [AdminInitiateAuth](#)
2. [AdminRespondToAuthChallenge](#)

`AdminInitiateAuth` dan `AdminRespondToAuthChallenge` memerlukan kredensial IAM dan cocok untuk klien aplikasi rahasia sisi server.

Pengguna mengautentikasi dengan menjawab tantangan berturut-turut hingga otentikasi gagal atau Amazon Cognito mengeluarkan token kepada pengguna. Anda dapat mengulangi langkah-langkah ini dengan Amazon Cognito, dalam proses yang mencakup berbagai tantangan, untuk mendukung alur otentikasi khusus apa pun.





Biasanya, aplikasi Anda menghasilkan prompt untuk mengumpulkan informasi dari pengguna, dan mengirimkan informasi tersebut dalam permintaan API ke Amazon Cognito. Pertimbangkan

`InitiateAuth` alur di kumpulan pengguna tempat Anda telah mengonfigurasi pengguna Anda dengan otentikasi multi-faktor (MFA).

1. Aplikasi Anda meminta pengguna Anda untuk nama pengguna dan kata sandi mereka.
2. Anda menyertakan nama pengguna dan kata sandi sebagai parameter di `InitiateAuth`.
3. Amazon Cognito mengembalikan `SMS_MFA` tantangan dan pengenal sesi.
4. Aplikasi Anda meminta pengguna Anda untuk kode MFA dari ponsel mereka.
5. Anda menyertakan kode itu dan pengenal sesi dalam `RespondToAuthChallenge` permintaan.

Bergantung pada fitur kumpulan pengguna, Anda dapat merespons beberapa tantangan `InitiateAuth` sebelum aplikasi mengambil token dari Amazon Cognito. Amazon Cognito menyertakan string sesi dalam menanggapi setiap permintaan. Untuk menggabungkan permintaan API Anda ke dalam alur otentikasi, sertakan string sesi dari respons ke permintaan sebelumnya di setiap permintaan berikutnya. Secara default, pengguna Anda memiliki waktu tiga menit untuk menyelesaikan setiap tantangan sebelum string sesi berakhir. Untuk menyesuaikan periode ini, ubah durasi sesi `Authentication client` aplikasi Anda. Prosedur berikut menjelaskan cara mengubah setelan ini dalam konfigurasi klien aplikasi Anda.

#### Note

Pengaturan durasi sesi alur otentikasi berlaku untuk autentikasi dengan API kumpulan pengguna Amazon Cognito. UI yang dihosting Amazon Cognito menetapkan durasi sesi menjadi 3 menit untuk otentikasi multi-faktor dan 8 menit untuk kode pengaturan ulang kata sandi.

## Amazon Cognito console

Untuk mengonfigurasi durasi sesi alur otentikasi klien aplikasi ( )AWS Management Console

1. Dari tab Integrasi aplikasi di kumpulan pengguna, pilih nama klien aplikasi Anda dari klien Aplikasi dan wadah analitik.
2. Pilih Edit di wadah informasi klien App.
3. Ubah nilai durasi sesi aliran otentikasi ke durasi validitas yang Anda inginkan, dalam hitungan menit, untuk kode MFA SMS. Ini juga mengubah jumlah waktu yang dimiliki pengguna untuk menyelesaikan tantangan otentikasi apa pun di klien aplikasi Anda.

#### 4. Pilih Simpan perubahan.

### Amazon Cognito API

Untuk mengonfigurasi durasi sesi alur otentikasi klien aplikasi (Amazon Cognito API)

1. Siapkan `UpdateUserPoolClient` permintaan dengan pengaturan kumpulan pengguna Anda yang ada dari `DescribeUserPoolClient` permintaan. `UpdateUserPoolClient` permintaan Anda harus menyertakan semua properti klien aplikasi yang ada.
2. Ubah nilai `AuthSessionValidity` ke durasi validitas yang Anda inginkan, dalam hitungan menit, untuk kode SMS MFA. Ini juga mengubah jumlah waktu yang dimiliki pengguna untuk menyelesaikan tantangan otentikasi apa pun di klien aplikasi Anda.

Untuk informasi selengkapnya tentang klien aplikasi, lihat [Klien aplikasi kumpulan pengguna](#).

Anda dapat menggunakan AWS Lambda pemicu untuk menyesuaikan cara pengguna mengotentikasi. Pemicu ini mengeluarkan dan memverifikasi tantangan mereka sendiri sebagai bagian dari alur autentikasi.

Anda juga dapat menggunakan alur otentikasi admin untuk server backend yang aman. Anda dapat menggunakan alur autentikasi migrasi pengguna untuk memungkinkan migrasi pengguna tanpa mengharuskan pengguna untuk mengatur ulang kata sandi mereka.

### Perilaku penguncian Amazon Cognito untuk upaya masuk yang gagal

Setelah lima upaya masuk yang tidak diautentikasi atau diautentikasi oleh IAM gagal dengan kata sandi, Amazon Cognito mengunci pengguna Anda selama satu detik. Durasi penguncian kemudian berlipat ganda setelah setiap upaya tambahan gagal, hingga maksimum sekitar 15 menit. Upaya yang dilakukan selama periode penguncian menghasilkan `Password attempts exceeded` pengecualian, dan tidak memengaruhi durasi periode penguncian berikutnya. Untuk jumlah kumulatif upaya masuk yang gagal  $n$ , tidak termasuk `Password attempts exceeded` pengecualian, Amazon Cognito mengunci pengguna Anda selama  $2^{(n-5)}$  detik. Untuk mengatur ulang penguncian ke status awal  $n=0$ , pengguna Anda harus berhasil masuk setelah periode penguncian berakhir, atau tidak memulai upaya masuk selama 15 menit berturut-turut kapan saja setelah penguncian. Perilaku ini dapat berubah. Perilaku ini tidak berlaku untuk tantangan khusus kecuali mereka juga melakukan otentikasi berbasis kata sandi.

## Topik

- [Alur otentikasi sisi klien](#)
- [Alur otentikasi sisi server](#)
- [Alur otentikasi kustom](#)
- [Alur otentikasi bawaan dan tantangan](#)
- [Alur otentikasi kustom dan tantangan](#)
- [Gunakan verifikasi kata sandi SRP dalam alur otentikasi khusus](#)
- [Alur autentikasi admin](#)
- [Alur autentikasi migrasi pengguna](#)

## Alur otentikasi sisi klien

[Proses berikut berfungsi untuk aplikasi sisi klien pengguna yang Anda buat dengan AWS Amplify atau SDK AWS .](#)

1. Pengguna memasukkan nama pengguna dan kata sandi mereka ke dalam aplikasi.
2. Aplikasi memanggil `InitiateAuth` operasi dengan nama pengguna dan detail Secure Remote Password (SRP) pengguna.

Operasi API ini mengembalikan parameter otentikasi.

### Note

Aplikasi ini menghasilkan detail SRP dengan fitur Amazon Cognito SRP yang dibangun ke SDK. AWS

3. Aplikasi ini memanggil operasi `RespondToAuthChallenge`. Jika panggilan berhasil, Amazon Cognito mengembalikan token pengguna, dan alur otentikasi selesai.

Jika Amazon Cognito memerlukan tantangan lain, panggilan untuk tidak `RespondToAuthChallenge` mengembalikan token. Sebaliknya, panggilan mengembalikan sesi.

4. Jika `RespondToAuthChallenge` mengembalikan sesi, aplikasi memanggil `RespondToAuthChallenge` lagi, kali ini dengan sesi dan respons tantangan (misalnya, kode MFA).

## Alur otentikasi sisi server

Jika Anda tidak memiliki aplikasi pengguna, tetapi Anda menggunakan backend aman Java, Ruby, atau Node.js atau aplikasi sisi server, Anda dapat menggunakan API sisi server yang diautentikasi untuk kumpulan pengguna Amazon Cognito.

Untuk aplikasi sisi server, autentikasi kumpulan pengguna mirip dengan otentikasi untuk aplikasi sisi klien, kecuali yang berikut ini:

- Aplikasi sisi server memanggil operasi API `AdminInitiateAuth` (bukan dari `InitiateAuth`). Operasi ini memerlukan AWS kredensial dengan izin yang menyertakan `dan. cognito-idp:AdminInitiateAuth` `cognito-idp:AdminRespondToAuthChallenge` Operasi mengembalikan parameter otentikasi yang diperlukan.
- Setelah aplikasi sisi server memiliki parameter otentikasi, aplikasi ini memanggil operasi `AdminRespondToAuthChallenge` API (bukan). `RespondToAuthChallenge` Operasi `AdminRespondToAuthChallenge` API hanya berhasil jika Anda memberikan AWS kredensial.

Untuk informasi selengkapnya tentang penandatanganan permintaan Amazon Cognito API dengan AWS kredensial, lihat [proses penandatanganan Versi Tanda Tangan 4](#) di Referensi Umum.AWS

Operasi `AdminInitiateAuth` dan `AdminRespondToAuthChallenge` API tidak dapat menerima kredensi `username-and-password` pengguna untuk login admin, kecuali jika Anda secara eksplisit mengaktifkannya untuk melakukannya dengan salah satu cara berikut:

- Sertakan `ALLOW_ADMIN_USER_PASSWORD_AUTH` (sebelumnya dikenal sebagai `ADMIN_NO_SRP_AUTH`) dalam `ExplicitAuthFlow` parameter saat Anda memanggil `CreateUserPoolClient` atau `UpdateUserPoolClient`
- Tambahkan `ALLOW_ADMIN_USER_PASSWORD_AUTH` ke daftar alur Autentikasi untuk klien aplikasi Anda. Konfigurasi klien aplikasi di tab Integrasi aplikasi di kumpulan pengguna Anda, di bawah Klien aplikasi dan analitik. Untuk informasi selengkapnya, lihat [Klien aplikasi kumpulan pengguna](#).

## Alur otentikasi kustom

Kumpulan pengguna Amazon Cognito juga memungkinkan untuk menggunakan alur otentikasi khusus, yang dapat membantu Anda membuat model otentikasi berbasis tantangan/respons menggunakan pemicu. AWS Lambda

**Note**

Anda tidak dapat menggunakan fitur keamanan lanjutan untuk kredensi yang disusupi dan autentikasi adaptif dengan alur otentikasi kustom. Untuk informasi selengkapnya, lihat [Menambahkan keamanan tingkat lanjut ke kumpulan pengguna](#).

Alur otentikasi khusus memungkinkan siklus tantangan dan respons yang disesuaikan untuk memenuhi persyaratan yang berbeda. Alur dimulai dengan panggilan ke operasi `InitiateAuth` API yang menunjukkan jenis otentikasi yang akan digunakan dan menyediakan parameter otentikasi awal apa pun. Amazon Cognito menanggapi `InitiateAuth` panggilan dengan salah satu jenis informasi berikut:

- Tantangan bagi pengguna, bersama dengan sesi dan parameter.
- Kesalahan jika pengguna gagal untuk mengautentikasi.
- ID, akses, dan refresh token jika parameter yang disediakan dalam `InitiateAuth` panggilan cukup untuk menandatangani pengguna. (Biasanya pengguna atau aplikasi harus terlebih dahulu menjawab tantangan, tetapi kode kustom Anda harus menentukan ini.)

Jika Amazon Cognito menanggapi `InitiateAuth` panggilan dengan tantangan, aplikasi mengumpulkan lebih banyak input dan memanggil operasi `RespondToAuthChallenge`. Panggilan ini memberikan tanggapan tantangan dan meneruskannya kembali sesi. Amazon Cognito menanggapi panggilan yang mirip dengan `RespondToAuthChallenge` panggilan tersebut. `InitiateAuth` Jika pengguna telah masuk, Amazon Cognito menyediakan token, atau jika pengguna tidak masuk, Amazon Cognito memberikan tantangan lain, atau kesalahan. Jika Amazon Cognito menampilkan tantangan lain, urutan akan berulang dan aplikasi akan memanggil `RespondToAuthChallenge` hingga pengguna berhasil masuk atau kesalahan dikembalikan. Untuk detail selengkapnya tentang operasi `InitiateAuth` dan `RespondToAuthChallenge` API, lihat [dokumentasi API](#).

## Alur otentikasi bawaan dan tantangan

Amazon Cognito berisi built-in `AuthFlow` dan `ChallengeName` nilai sehingga alur otentikasi standar dapat memvalidasi nama pengguna dan kata sandi melalui protokol Secure Remote Password (SRP). AWS SDK memiliki dukungan bawaan untuk aliran ini dengan Amazon Cognito.

Aliran dimulai dengan mengirim `USER_SRP_AUTH` sebagai `AuthFlow` ke `InitiateAuth`. Anda juga mengirim `USERNAME` dan `SRP_A` nilai `masukAuthParameters`. Jika `InitiateAuth` panggilan berhasil, respons telah dimasukkan `PASSWORD_VERIFIER` sebagai `ChallengeName` dan `SRP_B` dalam parameter tantangan. Aplikasi kemudian memanggil `RespondToAuthChallenge` dengan `PASSWORD_VERIFIER` `ChallengeName` dan parameter yang diperlukan dalam `ChallengeResponses`. Jika panggilan ke `RespondToAuthChallenge` berhasil dan pengguna masuk, Amazon Cognito mengeluarkan token. Jika Anda mengaktifkan otentikasi multi-faktor (MFA) untuk pengguna, Amazon Cognito mengembalikan dari. `ChallengeName` `SMS_MFA` Aplikasi ini dapat memberikan kode yang diperlukan melalui panggilan lain ke `RespondToAuthChallenge`.

## Alur otentikasi kustom dan tantangan

Aplikasi dapat memulai alur autentikasi kustom dengan memanggil `InitiateAuth` dengan `CUSTOM_AUTH` sebagai `AuthFlow`. Dengan alur otentikasi khusus, tiga Lambda memicu tantangan kontrol dan verifikasi tanggapan.

- Pemicu `DefineAuthChallenge` Lambda menggunakan array sesi tantangan dan respons sebelumnya sebagai masukan. Kemudian menghasilkan nama tantangan berikutnya dan Booleans yang menunjukkan apakah pengguna diautentikasi dan dapat diberikan token. Pemicu Lambda ini adalah mesin status yang mengontrol jalur pengguna melalui tantangan.
- Pemicu `CreateAuthChallenge` Lambda mengambil nama tantangan sebagai input dan menghasilkan tantangan dan parameter untuk mengevaluasi respons. Ketika `DefineAuthChallenge` kembali `CUSTOM_CHALLENGE` sebagai tantangan berikutnya, alur otentikasi memanggil `CreateAuthChallenge`. Pemicu `CreateAuthChallenge` Lambda melewati jenis tantangan berikutnya dalam parameter metadata tantangan.
- Fungsi Lambda `VerifyAuthChallengeResponse` mengevaluasi respons dan mengembalikan Boolean untuk menunjukkan apakah respons itu valid.

Alur otentikasi khusus juga dapat menggunakan kombinasi tantangan bawaan, seperti verifikasi kata sandi SRP dan MFA melalui SMS. Hal ini dapat menggunakan tantangan kustom seperti CAPTCHA atau pertanyaan rahasia.

## Gunakan verifikasi kata sandi SRP dalam alur otentikasi khusus

Jika Anda ingin menyertakan SRP dalam alur otentikasi khusus, Anda harus mulai dengan SRP.

- Untuk memulai verifikasi kata sandi SRP dalam aliran kustom, aplikasi akan memanggil `InitiateAuth` dengan `CUSTOM_AUTH` sebagai `Authflow`. Di `AuthParameters` peta, permintaan dari aplikasi Anda menyertakan `SRP_A`: (nilai SRP A) dan `CHALLENGE_NAME`: `SRP_A`.
- `CUSTOM_AUTH` aliran memanggil pemicu `DefineAuthChallenge` Lambda dengan sesi `challengeName`: `SRP_A` awal dan `challengeResult`: `true` Fungsi Lambda Anda merespons dengan `challengeName`: `PASSWORD_VERIFIER`, `issueTokens`: `false` dan `failAuthentication`: `false`
- Aplikasi selanjutnya harus memanggil `RespondToAuthChallenge` dengan `challengeName`: `PASSWORD_VERIFIER` dan parameter lain yang diperlukan untuk SRP di `challengeResponses` peta.
- Jika Amazon Cognito memverifikasi kata sandi, `RespondToAuthChallenge` panggil pemicu `DefineAuthChallenge` Lambda dengan sesi kedua dan `challengeName`: `PASSWORD_VERIFIER` `challengeResult`: `true` Pada saat itu, pemicu `DefineAuthChallenge` Lambda merespons dengan `challengeName`: `CUSTOM_CHALLENGE` untuk memulai tantangan khusus.
- Jika MFA diaktifkan untuk pengguna, setelah Amazon Cognito memverifikasi kata sandi, pengguna Anda kemudian ditantang untuk mengatur atau masuk dengan MFA.

#### Note

Halaman web masuk yang dihosting Amazon Cognito tidak dapat diaktifkan. [Tantangan otentikasi kustom pemicu Lambda](#)

Untuk informasi lebih lanjut tentang pemicu Lambda, termasuk kode sampel, lihat [Menyesuaikan alur kerja kumpulan pengguna dengan pemicu Lambda](#).

## Alur autentikasi admin

Praktik terbaik untuk otentikasi adalah menggunakan operasi API yang dijelaskan [Alur otentikasi kustom](#) dengan SRP untuk verifikasi kata sandi. AWS SDK menggunakan pendekatan itu, dan pendekatan ini membantu mereka menggunakan SRP. Namun, jika Anda ingin menghindari perhitungan SRP, serangkaian operasi API admin alternatif tersedia untuk server backend yang aman. Untuk implementasi admin backend ini, gunakan `AdminInitiateAuth` sebagai pengganti `InitiateAuth` Juga, gunakan `AdminRespondToAuthChallenge` sebagai pengganti `RespondToAuthChallenge`. Karena Anda dapat mengirimkan password sebagai



plaintext, Anda tidak perlu melakukan perhitungan SRP ketika Anda menggunakan operasi ini. Inilah contohnya:

```
AdminInitiateAuth Request {
  "AuthFlow": "ADMIN_USER_PASSWORD_AUTH",
  "AuthParameters": {
    "USERNAME": "<username>",
    "PASSWORD": "<password>"
  },
  "ClientId": "<clientId>",
  "UserPoolId": "<userPoolId>"
}
```

Operasi autentikasi admin ini memerlukan kredensial developer dan menggunakan proses penandatanganan AWS Signature Version 4 (SiGv4). Operasi ini tersedia dalam AWS SDK standar, termasuk Node.js, yang nyaman untuk fungsi Lambda. Untuk menggunakan operasi ini dan meminta mereka menerima kata sandi dalam teks biasa, Anda harus mengaktifkannya untuk aplikasi di konsol. Sebagai alternatif, Anda dapat melewati parameter `ADMIN_USER_PASSWORD_AUTH` untuk `ExplicitAuthFlow` dalam panggilan ke `CreateUserPoolClient` atau `UpdateUserPoolClient`. `RespondToAuthChallenge` Operasi `InitiateAuth` dan tidak menerima `ADMIN_USER_PASSWORD_AUTH` `AuthFlow`.

Dalam `AdminInitiateAuth` `tanggapanChallengeParameters`, `USER_ID_FOR_SRP` atribut, jika ada, berisi nama pengguna aktual pengguna, bukan alias (seperti alamat email atau nomor telepon). Dalam panggilan Anda ke `AdminRespondToAuthChallenge`, di `ChallengeResponses`, Anda harus meneruskan nama pengguna ini di `USERNAME` parameter.

#### Note

Karena implementasi admin backend menggunakan alur otentikasi admin, alur tidak mendukung pelacakan perangkat. Saat Anda mengaktifkan pelacakan perangkat, otentikasi admin berhasil, tetapi panggilan apa pun untuk menyegarkan token akses gagal.

## Alur autentikasi migrasi pengguna

Pemicu Lambda migrasi pengguna membantu memigrasikan pengguna dari sistem manajemen pengguna lama ke kumpulan pengguna Anda. Jika Anda memilih alur `USER_PASSWORD_AUTH` otentikasi, pengguna tidak perlu mengatur ulang kata sandi mereka selama migrasi pengguna. Alur

ini mengirimkan kata sandi pengguna Anda ke layanan melalui koneksi SSL terenkripsi selama autentikasi.

Jika Anda telah memigrasikan semua pengguna, alihkan aliran ke aliran SRP yang lebih aman. Aliran SRP tidak mengirim kata sandi apa pun melalui jaringan.

Untuk mempelajari lebih lanjut tentang pemicu Lambda, lihat [Menyesuaikan alur kerja kumpulan pengguna dengan pemicu Lambda](#)

Untuk informasi selengkapnya tentang memigrasi pengguna dengan pemicu Lambda, lihat [Mengimpor pengguna ke kumpulan pengguna dengan migrasi pengguna pemicu Lambda](#)

## Klien aplikasi kumpulan pengguna

Klien aplikasi kumpulan pengguna adalah konfigurasi dalam kumpulan pengguna yang berinteraksi dengan satu aplikasi seluler atau web yang mengautentikasi dengan Amazon Cognito. Klien aplikasi dapat memanggil operasi API yang diautentikasi dan tidak diautentikasi, dan membaca atau memodifikasi beberapa atau semua atribut pengguna Anda. Aplikasi Anda harus mengidentifikasi dirinya ke klien aplikasi dalam operasi untuk mendaftar, masuk, dan menangani kata sandi yang terlupakan. Permintaan API ini harus menyertakan identifikasi diri dengan ID klien aplikasi, dan otorisasi dengan rahasia klien opsional. Anda harus mengamankan ID atau rahasia klien aplikasi apa pun sehingga hanya aplikasi klien resmi yang dapat memanggil operasi yang tidak diautentikasi ini. Selain itu, jika mengonfigurasi aplikasi untuk menandatangani permintaan API yang diautentikasi dengan AWS kredensial, Anda harus mengamankan kredensialnya terhadap pemeriksaan pengguna.

Anda dapat membuat beberapa aplikasi untuk sebuah kolam pengguna. Klien aplikasi mungkin ditautkan ke platform kode aplikasi, atau penyewa terpisah di kumpulan pengguna Anda. Misalnya, Anda dapat membuat aplikasi untuk aplikasi sisi server dan aplikasi Android yang berbeda. Setiap aplikasi memiliki ID klien aplikasi sendiri.

## Jenis klien aplikasi

Saat membuat klien aplikasi di Amazon Cognito, Anda dapat mengisi opsi terlebih dahulu berdasarkan klien publik tipe klien OAuth standar dan klien rahasia. Konfigurasikan klien rahasia dengan rahasia klien. Untuk informasi selengkapnya tentang tipe klien, lihat [IETF RFC 6749 #2 .1](#).

### Klien publik

Klien publik berjalan di browser atau di perangkat seluler. Karena tidak memiliki sumber daya sisi server tepercaya, ia tidak memiliki rahasia klien.

## Klien rahasia

Klien rahasia memiliki sumber daya sisi server yang dapat dipercaya dengan rahasia klien untuk operasi API yang tidak diautentikasi. Aplikasi ini mungkin berjalan sebagai skrip daemon atau shell di server backend Anda.

## Rahasia klien

Rahasia klien, atau kata sandi klien, adalah string tetap yang harus digunakan aplikasi Anda di semua permintaan API ke klien aplikasi. Klien aplikasi Anda harus memiliki rahasia klien untuk melakukan `client_credentials` hibah. Untuk informasi selengkapnya, lihat [IETF RFC 6749 #2 .3.1](#).

Anda tidak dapat mengubah rahasia setelah membuat aplikasi. Anda dapat membuat aplikasi baru dengan rahasia baru jika Anda ingin memutar rahasia. Anda juga dapat menghapus aplikasi untuk memblokir akses dari aplikasi yang menggunakan ID klien aplikasi tersebut.

Anda dapat menggunakan klien rahasia, dan rahasia klien, dengan aplikasi publik. Gunakan CloudFront proxy Amazon untuk menambahkan `SECRET_HASH` dalam perjalanan. Untuk informasi selengkapnya, lihat [Melindungi klien publik untuk Amazon Cognito dengan menggunakan CloudFront proxy Amazon](#) di AWS blog.

## Token web JSON

Klien aplikasi Amazon Cognito dapat mengeluarkan token web JSON (JWT) dari jenis berikut.

### Token identitas (ID)

Pernyataan yang dapat diverifikasi bahwa pengguna Anda diautentikasi dari kumpulan pengguna Anda. OpenID Connect (OIDC) menambahkan [spesifikasi token ID ke standar akses dan penyegaran token](#) yang ditentukan oleh OAuth 2.0. Token ID berisi informasi identitas, seperti atribut pengguna, yang dapat digunakan aplikasi Anda untuk membuat profil pengguna dan sumber daya penyediaan. Untuk informasi selengkapnya, lihat [Menggunakan token ID](#).

### Token akses

Pernyataan hak akses pengguna Anda yang dapat diverifikasi. Token akses berisi [cakupan](#), fitur OIDC dan OAuth 2.0. Aplikasi Anda dapat menampilkan cakupan ke sumber daya back-end dan membuktikan bahwa kumpulan pengguna Anda mengizinkan pengguna atau mesin untuk mengakses data dari API, atau data pengguna mereka sendiri. Token akses dengan cakupan

khusus, seringkali dari hibah kredensial klien M2M, mengotorisasi akses ke server sumber daya. Untuk informasi selengkapnya, lihat [Menggunakan token akses](#).

## Token refresh

Pernyataan otentikasi awal terenkripsi yang dapat ditampilkan aplikasi Anda ke kumpulan pengguna Anda saat token pengguna Anda kedaluwarsa. Permintaan refresh token mengembalikan akses baru dan token ID yang belum kedaluwarsa. Untuk informasi selengkapnya, lihat [Menggunakan token penyegaran](#).

Anda dapat mengatur kedaluwarsa token ini untuk setiap klien aplikasi dari tab Integrasi aplikasi kumpulan pengguna Anda di konsol [Amazon Cognito](#).

## Ketentuan klien aplikasi

Istilah berikut adalah properti klien aplikasi yang tersedia di konsol Amazon Cognito.

### URL callback yang diizinkan

URL callback menunjukkan tempat pengguna akan dialihkan setelah login berhasil. Pilih setidaknya satu URL panggilan balik. URL callback harus:

- URI absolut.
- Sudah terdaftar pada klien.
- Tidak termasuk komponen fragmen.

Lihat [OAuth 2.0 - titik akhir pengalihan](#).

Amazon Cognito membutuhkan HTTPS di atas HTTP kecuali `http://localhost` untuk tujuan pengujian saja.

URL callback aplikasi seperti `myapp://example` juga didukung.

### URL keluar yang diizinkan

URL keluar menunjukkan ke mana pengguna Anda akan dialihkan setelah keluar.

### Atribut izin baca dan tulis

Kumpulan pengguna Anda mungkin memiliki banyak pelanggan, masing-masing dengan klien aplikasi mereka sendiri dan IdPs. Anda dapat mengonfigurasi klien aplikasi agar memiliki akses

baca dan tulis hanya ke atribut pengguna yang relevan dengan aplikasi. Dalam kasus seperti otorisasi machine-to-machine (M2M), Anda tidak dapat memberikan akses ke atribut pengguna Anda.

Pertimbangan untuk konfigurasi izin baca dan tulis atribut

- Saat Anda membuat klien aplikasi dan tidak menyesuaikan izin baca dan tulis atribut, Amazon Cognito memberikan izin baca dan tulis ke semua atribut kumpulan pengguna.
- Anda dapat memberikan akses tulis ke atribut [kustom](#) yang tidak dapat diubah. Klien aplikasi Anda dapat menulis nilai ke atribut yang tidak dapat diubah saat Anda membuat atau mendaftarkan pengguna. Setelah ini, Anda tidak dapat menulis nilai ke atribut kustom yang tidak dapat diubah untuk pengguna.
- Klien aplikasi harus memiliki akses tulis ke atribut yang diperlukan di kumpulan pengguna Anda. Konsol Amazon Cognito secara otomatis menetapkan atribut yang diperlukan sebagai dapat ditulis.
- Anda tidak dapat mengizinkan klien aplikasi untuk memiliki akses tulis ke `email_verified` atau `phone_number_verified`. Administrator kumpulan pengguna dapat memodifikasi nilai-nilai ini. Seorang pengguna hanya dapat mengubah nilai atribut ini melalui [verifikasi atribut](#).

Alur otentikasi

Metode yang memungkinkan klien aplikasi Anda untuk login. Aplikasi Anda dapat mendukung otentikasi dengan nama pengguna dan kata sandi, Secure Remote Password (SRP), autentikasi khusus dengan pemicu Lambda, dan penyegaran token. Sebagai praktik keamanan terbaik, gunakan otentikasi SRP sebagai metode masuk utama Anda. UI yang dihosting secara otomatis menandatangani pengguna dengan SRP.

Lingkup kustom

Cakupan kustom adalah salah satu yang Anda tetapkan untuk server sumber daya Anda sendiri di Server Sumber Daya. Formatnya adalah *resource-server-identifier/scope*. Lihat [Cakupan, M2M, dan otorisasi API dengan server sumber daya](#).

URI pengalihan default

Mengganti `redirect_uri` parameter dalam permintaan otentikasi untuk pengguna dengan pihak ketiga. IdPs Konfigurasi setelan klien aplikasi ini dengan `DefaultRedirectURI` parameter permintaan [CreateUserPoolClient](#) atau [UpdateUserPoolClient](#) API. URL ini juga harus menjadi anggota klien aplikasi Anda. `CallbackURLs` Amazon Cognito mengalihkan sesi yang diautentikasi ke URL ini saat:

1. Klien aplikasi Anda memiliki satu [penyedia identitas](#) yang ditetapkan dan beberapa [URL callback](#) yang ditentukan. Kumpulan pengguna Anda mengalihkan permintaan otentikasi ke [server otorisasi](#) ke URI pengalihan default jika tidak menyertakan parameter. `redirect_uri`
2. Klien aplikasi Anda memiliki satu [penyedia identitas](#) yang ditetapkan dan satu [URL callback](#) ditentukan. Dalam skenario ini, tidak perlu mendefinisikan URL callback default. Permintaan yang tidak menyertakan pengalihan `redirect_uri` parameter ke URL callback yang tersedia.

## Penyedia Identitas

Anda dapat memilih beberapa atau semua penyedia identitas eksternal kumpulan pengguna (IdPs) untuk mengautentikasi pengguna Anda. Klien aplikasi Anda juga dapat mengautentikasi hanya pengguna lokal di kumpulan pengguna Anda. Saat menambahkan iDP ke klien aplikasi, Anda dapat membuat tautan otorisasi ke iDP dan menampilkannya di halaman login UI yang dihosting. Anda dapat menetapkan beberapa IdPs, tetapi Anda harus menetapkan setidaknya satu. Untuk informasi selengkapnya tentang penggunaan eksternal IdPs, lihat [Menambahkan login kumpulan pengguna melalui pihak ketiga](#).

## Lingkup OpenID Connect

Pilih salah satu atau beberapa cakupan OAuth berikut untuk menentukan hak akses yang dapat diminta untuk token akses.

- `openid`Cakupan menyatakan bahwa Anda ingin mengambil token ID dan ID unik pengguna. Ini juga meminta semua atau beberapa atribut pengguna, tergantung pada cakupan tambahan dalam permintaan. Amazon Cognito tidak mengembalikan token ID kecuali Anda meminta cakupannya. `openid` `openid`Cakupan mengotorisasi klaim token ID struktural seperti kedaluwarsa dan ID kunci, dan menentukan atribut pengguna yang Anda terima sebagai respons dari. [Titik akhir UserInfo](#)
- `openid`Kapan satu-satunya cakupan yang Anda minta, Amazon Cognito mengisi token ID dengan semua atribut pengguna yang dapat dibaca oleh klien aplikasi saat ini. `userInfo`Respons terhadap token akses dengan cakupan ini saja mengembalikan semua atribut pengguna.
- Ketika Anda meminta `openid` dengan cakupan lain seperti `phoneemail`, `atauprofile`, token ID dan `userInfo` mengembalikan ID unik pengguna dan atribut yang ditentukan oleh cakupan tambahan.
- Cakupan `phone` memberikan akses ke `phone_number` dan klaim `phone_number_verified`. Cakupan ini hanya dapat diminta dengan cakupan `openid`.
- Cakupan `email` memberikan akses ke `email` dan klaim `email_verified`. Cakupan ini hanya dapat diminta dengan cakupan `openid`.

- Ruang `aws.cognito.signin.user.admin` lingkup memberikan akses ke operasi [API kumpulan pengguna Amazon Cognito](#) yang memerlukan token akses, seperti [UpdateUserAttributes](#) dan [VerifyUserAttribute](#)
- Cakupan `profile` memberikan akses ke semua atribut pengguna yang dapat dibaca oleh klien. Cakupan ini hanya dapat diminta dengan cakupan `openid`.

Untuk informasi selengkapnya tentang cakupan, lihat daftar cakupan [OIDC standar](#).

## Jenis hibah OAuth

Hibah OAuth adalah metode otentikasi yang mengambil token kumpulan pengguna. Amazon Cognito mendukung jenis hibah berikut. Untuk mengintegrasikan hibah OAuth ini di aplikasi Anda, Anda harus menambahkan domain ke kumpulan pengguna Anda.

### Pemberian kode otorisasi

Pemberian kode otorisasi menghasilkan kode yang dapat ditukarkan oleh aplikasi Anda dengan token kumpulan pengguna dengan [Titik akhir token](#). Saat Anda menukar kode otorisasi, aplikasi Anda akan menerima ID, akses, dan token penyegaran. Alur OAuth ini, seperti hibah implisit, terjadi di browser pengguna Anda. Pemberian kode otorisasi adalah hibah paling aman yang ditawarkan Amazon Cognito, karena token tidak terlihat di sesi pengguna Anda. Sebagai gantinya, aplikasi Anda menghasilkan permintaan yang mengembalikan token dan dapat menyimpannya di penyimpanan yang dilindungi. Untuk informasi selengkapnya, lihat Kode otorisasi di [IETF RFC 6749 #1 .3.1](#)

#### Note

Sebagai praktik keamanan terbaik di aplikasi klien publik, aktifkan hanya aliran OAuth pemberian kode otorisasi, dan terapkan Kunci Bukti untuk Pertukaran Kode (PKCE) untuk membatasi pertukaran token. Dengan PKCE, klien hanya dapat bertukar kode otorisasi ketika mereka telah memberikan titik akhir token dengan rahasia yang sama yang disajikan dalam permintaan otentikasi asli. Untuk informasi lebih lanjut tentang PKCE, lihat [IETF RFC 7636](#).

### Hibah implisit

Hibah implisit memberikan akses dan token ID, tetapi bukan token penyegaran, ke sesi browser pengguna Anda langsung dari [Otorisasi titik akhir](#). Hibah implisit menghapus persyaratan

untuk permintaan terpisah ke titik akhir token, tetapi tidak kompatibel dengan PKCE dan tidak mengembalikan token penyegaran. Hibah ini mengakomodasi skenario pengujian dan arsitektur aplikasi yang tidak dapat menyelesaikan hibah kode otorisasi. Untuk informasi selengkapnya, lihat Hibah implisit di [IETF RFC 6749 #1 .3.2](#). Anda dapat mengaktifkan hibah kode otorisasi dan hibah implisit di klien aplikasi, lalu menggunakan setiap hibah sesuai kebutuhan.

## Pemberian kredensial klien

Hibah kredensial klien adalah untuk komunikasi machine-to-machine (M2M). Kode otorisasi dan implisit memberikan token penerbitan kepada pengguna manusia yang diautentikasi. Kredensial klien memberikan otorisasi berbasis cakupan dari sistem non-interaktif ke API. Aplikasi Anda dapat meminta kredensial klien langsung dari titik akhir token dan menerima token akses. Untuk informasi selengkapnya, lihat Client Client Client di [IETF RFC 6749 #1 .3.4](#). Anda hanya dapat mengaktifkan hibah kredensial klien di klien aplikasi yang memiliki rahasia klien dan yang tidak mendukung kode otorisasi atau hibah implisit.

### Note

Karena Anda tidak memanggil alur kredensial klien sebagai pengguna, hibah ini hanya dapat menambahkan cakupan khusus untuk mengakses token. Cakupan kustom adalah salah satu yang Anda tetapkan untuk server sumber daya Anda sendiri. Cakupan default menyukai `openid` dan `profile` tidak berlaku untuk pengguna bukan manusia.

Karena token ID adalah validasi atribut pengguna, token tersebut tidak relevan dengan komunikasi M2M, dan pemberian kredensial klien tidak menerbitkannya. Lihat [Cakupan, M2M, dan otorisasi API dengan server sumber daya](#).

Hibah kredensi klien menambah biaya ke tagihan Anda. AWS Untuk informasi selengkapnya, lihat [Harga Amazon Cognito](#).

## Membuat klien aplikasi


### AWS Management Console

Untuk membuat klien aplikasi (konsol)

1. Masuk ke [Konsol Amazon Cognito](#). Jika diminta, masukkan AWS kredensial Anda.
2. Pilih Kolam Pengguna.



3. Pilih kolam pengguna yang ada dari daftar, atau buat kolam pengguna.
4. Pilih tab Integrasi aplikasi.
5. Di bawah Klien aplikasi, pilih Buat klien aplikasi.
6. Pilih jenis Aplikasi: Klien publik, Klien Rahasia, atau Lainnya.
7. Masukkan nama klien Aplikasi.
8. Pilih Hasilkan rahasia klien agar Amazon Cognito menghasilkan rahasia klien untuk Anda. Rahasia klien biasanya dikaitkan dengan klien rahasia.
9. Pilih alur Autentikasi yang ingin Anda izinkan di klien aplikasi Anda.
10. Konfigurasi durasi sesi alur Otentikasi. Ini adalah jumlah waktu yang dimiliki pengguna Anda untuk menyelesaikan setiap tantangan otentikasi sebelum token sesi mereka kedaluwarsa.
11. (Opsional) Jika Anda ingin mengonfigurasi kedaluwarsa token, selesaikan langkah-langkah berikut:
  - a. Tentukan kedaluwarsa token Refresh untuk klien aplikasi. Nilai default-nya adalah 30 hari. Anda dapat mengubahnya ke nilai berapa pun antara 1 jam dan 10 tahun.
  - b. Tentukan kedaluwarsa token Access untuk klien aplikasi. Nilai default adalah 1 jam. Anda dapat mengubahnya ke nilai berapa pun antara 5 menit dan 24 jam.
  - c. Tentukan kedaluwarsa token ID untuk klien aplikasi. Nilai default adalah 1 jam. Anda dapat mengubahnya ke nilai berapa pun antara 5 menit dan 24 jam.

 Important

Jika Anda menggunakan UI yang dihosting dan mengonfigurasi masa pakai token kurang dari satu jam, pengguna Anda akan dapat menggunakan token berdasarkan durasi cookie sesi mereka, yang saat ini diperbaiki pada satu jam.

12. Pilih apakah Anda akan Aktifkan pencabutan token untuk klien aplikasi ini. Ini akan meningkatkan ukuran token yang dikeluarkan Amazon Cognito.
13. Pilih apakah Anda akan Mencegah kesalahan keberadaan pengguna untuk klien aplikasi ini. Amazon Cognito akan menanggapi permintaan masuk untuk pengguna yang tidak ada dengan pesan umum yang menyatakan bahwa nama pengguna atau kata sandi salah.
14. Jika Anda ingin menggunakan UI yang dihosting dengan klien aplikasi ini, konfigurasi pengaturan UI yang Dihosting.

- a. Masukkan satu atau beberapa URL panggilan balik yang Diizinkan. Ini adalah URL web atau aplikasi tempat Anda ingin Amazon Cognito mengarahkan pengguna Anda setelah mereka menyelesaikan otentikasi.
  - b. Masukkan satu atau beberapa URL keluar yang diizinkan. Ini adalah URL yang Anda ingin aplikasi Anda terima dalam permintaan ke. [Titik akhir logout](#)
  - c. Pilih satu atau beberapa penyedia Identitas yang Anda inginkan agar dapat masuk pengguna untuk aplikasi Anda. Anda dapat memilih kombinasi apa pun yang ada IdPs. Anda dapat mengautentikasi pengguna dengan kumpulan pengguna Anda sendiri, atau dengan satu atau beberapa pihak ketiga IdPs yang Anda konfigurasi di kumpulan pengguna Anda.
  - d. Pilih jenis hibah OAuth 2.0 yang Anda ingin klien aplikasi Anda terima.
    - Pilih Pemberian kode otorisasi untuk meneruskan kode ke aplikasi Anda yang dapat ditukarkan dengan token dengan. [Titik akhir token](#)
    - Pilih Implicit grant untuk meneruskan ID dan mengakses token langsung ke aplikasi Anda. Alur hibah implisit mengekspos token langsung ke pengguna Anda.
    - Pilih kredensial Klien untuk meneruskan token akses ke aplikasi Anda berdasarkan pengetahuannya bukan tentang kredensial pengguna, tetapi rahasia klien. Alur hibah kredensial klien saling eksklusif dengan kode otorisasi dan aliran hibah implisit.
  - e. Pilih cakupan OpenID Connect yang ingin Anda otorisasi untuk digunakan dengan klien aplikasi Anda. Anda dapat menghasilkan token akses hanya dengan `aws.cognito.signin.user.admin` cakupan melalui API kumpulan pengguna. Untuk cakupan tambahan, Anda harus meminta token akses Anda dari. [Titik akhir token](#)
  - f. Pilih cakupan Kustom yang ingin Anda otorisasi dengan klien aplikasi Anda. Cakupan kustom paling sering digunakan untuk mengotorisasi akses ke API pihak ketiga.
15. Konfigurasi izin baca dan tulis Atribut untuk klien aplikasi ini. Klien aplikasi Anda dapat memiliki izin untuk membaca dan menulis semua, atau subset terbatas, skema atribut kumpulan pengguna Anda.
  16. Pilih Buat klien aplikasi.
  17. Perhatikan id Klien. Ini akan mengidentifikasi klien aplikasi dalam permintaan pendaftaran dan masuk.

## AWS CLI

```
aws cognito-idp create-user-pool-client --user-pool-id MyUserPoolID --client-name myApp
```

### Note

Gunakan format JSON untuk URL callback dan keluar agar CLI tidak memperlakukannya sebagai file parameter jarak jauh:

```
--callback-urls ["https://example.com"]  
--logout-urls ["https://example.com"]
```

Lihat referensi AWS CLI perintah untuk informasi lebih lanjut: [create-user-pool-client](#)

## Amazon Cognito user pools API

Hasilkan permintaan [CreateUserPoolClient](#) API. Anda harus menentukan nilai untuk semua parameter yang tidak ingin disetel ke nilai default.

## Memperbarui klien aplikasi kumpulan pengguna (AWS CLI dan AWS API)

Di AWS CLI, masukkan perintah berikut:

```
aws cognito-idp update-user-pool-client --user-pool-id "MyUserPoolID" --client-id "MyAppClientID" --allowed-o-auth-flows-user-pool-client --allowed-o-auth-flows "code" "implicit" --allowed-o-auth-scopes "openid" --callback-urls ["https://example.com"] --supported-identity-providers ["MySAMLIdP", "LoginWithAmazon"]
```

Jika perintah berhasil, AWS CLI mengembalikan konfirmasi:

```
{  
  "UserPoolClient": {  
    "ClientId": "MyClientID",  
    "SupportedIdentityProviders": [  
      "LoginWithAmazon",  
      "MySAMLIdP"  
    ],  
  },  
}
```

```
"CallbackURLs": [
  "https://example.com"
],
"AllowedOAuthScopes": [
  "openid"
],
"ClientName": "Example",
"AllowedOAuthFlows": [
  "implicit",
  "code"
],
"RefreshTokenValidity": 30,
"AuthSessionValidity": 3,
"CreationDate": 1524628110.29,
"AllowedOAuthFlowsUserPoolClient": true,
"UserPoolId": "MyUserPoolID",
"LastModifiedDate": 1530055177.553
}
}
```

Lihat referensi AWS CLI perintah untuk informasi lebih lanjut: [update-user-pool-client](#).

AWS API: [UpdateUserPoolClient](#)

Mendapatkan informasi tentang klien aplikasi kumpulan pengguna (AWS CLI dan AWS API)

```
aws cognito-idp describe-user-pool-client --user-pool-id MyUserPoolID --client-id MyClientID
```

Lihat referensi AWS CLI perintah untuk informasi lebih lanjut: [describe-user-pool-client](#).

AWS API: [DescribeUserPoolClient](#)

Mencantumkan semua informasi klien aplikasi di kumpulan pengguna (AWS CLI dan AWS API)

```
aws cognito-idp list-user-pool-clients --user-pool-id "MyUserPoolID" --max-results 3
```

Lihat referensi AWS CLI perintah untuk informasi lebih lanjut: [list-user-pool-clients](#).

AWS API: [ListUserPoolClients](#)

## Menghapus klien aplikasi kumpulan pengguna (AWS CLI dan AWS API)

```
aws cognito-idp delete-user-pool-client --user-pool-id "MyUserPoolID" --client-id "MyAppClientID"
```

Lihat referensi AWS CLI perintah untuk informasi lebih lanjut: [delete-user-pool-client](#)

AWS API: [DeleteUserPoolClient](#)

## Bekerja dengan perangkat pengguna di kumpulan pengguna Anda

Saat Anda masuk ke pengguna kumpulan pengguna lokal dengan API kumpulan pengguna Amazon Cognito, Anda dapat mengaitkan log aktivitas pengguna dari [fitur keamanan lanjutan](#) dengan masing-masing perangkat mereka dan, secara opsional, memungkinkan pengguna melewati otentikasi multi-faktor (MFA) jika mereka menggunakan perangkat tepercaya. Amazon Cognito menyertakan kunci perangkat sebagai respons terhadap proses masuk apa pun yang belum menyertakan informasi perangkat. Kunci perangkat ada dalam format *Region\_UUID*. Dengan kunci perangkat, library Secure Remote Password (SRP), dan kumpulan pengguna yang mengizinkan autentikasi perangkat, Anda dapat meminta pengguna di aplikasi untuk mempercayai perangkat saat ini dan tidak lagi meminta kode MFA saat masuk.

### Topik

- [Menyiapkan perangkat yang diingat](#)
- [Mendapatkan kunci perangkat](#)
- [Masuk dengan perangkat](#)
- [Melihat, memperbarui, dan melupakan perangkat](#)

## Menyiapkan perangkat yang diingat

Dengan kumpulan pengguna Amazon Cognito, Anda dapat mengaitkan setiap perangkat pengguna dengan pengenalan perangkat unik: kunci perangkat. Saat Anda menampilkan kunci perangkat dan melakukan otentikasi perangkat saat masuk, Anda dapat memanfaatkan dua fitur.

1. Dengan fitur keamanan canggih, Anda dapat memantau aktivitas pengguna pada perangkat tertentu untuk tujuan keamanan dan analitik. Saat pengguna masuk, aplikasi Anda memiliki

opsi untuk mengautentikasi setiap pengguna dan perangkat mereka, menambahkan informasi perangkat ke log aktivitas Anda.

2. Perangkat mengingat juga mendukung alur autentikasi perangkat tepercaya, di mana pengguna Anda dapat memilih untuk masuk tanpa MFA untuk jangka waktu yang sesuai dengan persyaratan keamanan aplikasi Anda. Saat Anda ingin meminta ulang pengguna Anda untuk mengirimkan kode MFA, Anda dapat mengubah status perangkat mereka yang diingat.

Perangkat yang diingat dapat mengganti MFA hanya di kumpulan pengguna dengan MFA aktif.

Saat pengguna masuk dengan perangkat yang diingat, Anda harus melakukan autentikasi perangkat tambahan selama alur autentikasi mereka. Untuk informasi selengkapnya, lihat [Masuk dengan perangkat](#).

Konfigurasi kumpulan pengguna Anda untuk mengingat perangkat di tab Pengalaman masuk di kumpulan pengguna Anda, di bawah Pelacakan perangkat. Saat menyiapkan fungsionalitas perangkat yang diingat melalui konsol Amazon Cognito, Anda memiliki tiga pilihan: Selalu, Pilihan Pengguna, dan Tidak.

#### Jangan ingat

Kumpulan pengguna Anda tidak meminta pengguna untuk mengingat perangkat saat mereka masuk.

#### Selalu ingat

Saat aplikasi mengonfirmasi perangkat pengguna, kumpulan pengguna akan selalu mengingat perangkat tersebut dan tidak menampilkan tantangan MFA pada proses login perangkat yang sukses di masa mendatang.

#### Keikutsertaan pengguna

Saat aplikasi mengonfirmasi perangkat pengguna, kumpulan pengguna tidak secara otomatis menekan tantangan MFA. Anda harus meminta pengguna Anda untuk memilih apakah mereka ingin mengingat perangkat.

Saat Anda memilih Selalu ingat atau Ikut serta Pengguna, Amazon Cognito menghasilkan kunci dan rahasia pengenalan perangkat setiap kali pengguna masuk dari perangkat yang tidak dikenal. Kunci perangkat adalah pengenalan awal yang dikirimkan aplikasi ke kumpulan pengguna saat pengguna melakukan autentikasi perangkat.

Dengan setiap perangkat pengguna yang dikonfirmasi, baik diingat secara otomatis atau ikut serta, Anda dapat menggunakan kunci dan rahasia pengenalan perangkat untuk mengautentikasi perangkat pada setiap pengguna yang masuk.

Anda juga dapat mengonfigurasi pengaturan perangkat yang diingat untuk kumpulan pengguna Anda dalam permintaan [CreateUserPool](#) atau [UpdateUserPool](#) API. Untuk informasi lebih lanjut, lihat [DeviceConfiguration](#) properti.

API kumpulan pengguna Amazon Cognito memiliki operasi tambahan untuk perangkat yang diingat.

1. [ListDevices](#) dan [AdminListDevices](#) mengembalikan daftar kunci perangkat dan metadatanya untuk pengguna.
2. [GetDevice](#) dan [AdminGetDevice](#) mengembalikan kunci perangkat dan metadata untuk satu perangkat.
3. [UpdateDeviceStatus](#) dan [AdminUpdateDeviceStatus](#) mengatur perangkat pengguna seperti yang diingat atau tidak diingat.
4. [ForgetDevice](#) dan [AdminForgetDevice](#) menghapus perangkat yang dikonfirmasi pengguna dari profil mereka.

Operasi API dengan nama yang dimulai untuk Admin digunakan dalam aplikasi sisi server dan harus diotorisasi dengan kredensi IAM. Untuk informasi selengkapnya, lihat [Menggunakan API kumpulan pengguna Amazon Cognito dan titik akhir kumpulan pengguna](#).

## Mendapatkan kunci perangkat

Kapan pun pengguna Anda masuk dengan API kumpulan pengguna dan tidak menyertakan kunci perangkat dalam parameter autentikasi `DEVICE_KEY`, Amazon Cognito mengembalikan kunci perangkat baru dalam respons. Di aplikasi sisi klien publik Anda, letakkan kunci perangkat di penyimpanan aplikasi sehingga Anda dapat memasukkannya ke dalam permintaan future. Di aplikasi sisi server rahasia Anda, atur cookie browser atau token sisi klien lainnya dengan kunci perangkat pengguna Anda.

Sebelum pengguna Anda dapat masuk dengan perangkat tepercaya mereka, aplikasi Anda harus mengonfirmasi kunci perangkat dan memberikan informasi tambahan. Buat [ConfirmDevice](#) permintaan ke Amazon Cognito yang mengonfirmasi perangkat pengguna Anda dengan kunci perangkat, nama yang ramah, pemverifikasi kata sandi, dan garam. Jika Anda mengonfigurasi kumpulan pengguna untuk autentikasi perangkat opt-in, Amazon Cognito

merespons permintaan `ConfirmDevice` Anda dengan prompt bahwa pengguna harus memilih apakah akan mengingat perangkat saat ini. Tanggapi dengan pilihan pengguna Anda dalam [UpdateDeviceStatus](#) permintaan.

Saat mengonfirmasi perangkat pengguna tetapi tidak menyetelnya seperti yang diingat, Amazon Cognito menyimpan asosiasi tersebut tetapi dilanjutkan dengan login non-perangkat saat Anda memberikan kunci perangkat. Perangkat dapat menghasilkan log yang berguna untuk keamanan pengguna dan pemecahan masalah. Perangkat yang dikonfirmasi tetapi tidak diingat tidak memanfaatkan fitur masuk, tetapi memanfaatkan fitur log pemantauan keamanan. Saat Anda mengaktifkan fitur keamanan lanjutan untuk klien aplikasi dan menyandikan jejak perangkat ke dalam permintaan Anda, Amazon Cognito mengaitkan peristiwa pengguna dengan perangkat yang dikonfirmasi.

Untuk mendapatkan kunci perangkat baru

1. Mulai sesi login pengguna Anda dengan permintaan [InitiateAuth](#) API.
2. Tanggapi semua tantangan otentikasi [RespondToAuthChallenge](#) hingga Anda menerima token web JSON (JWT) yang menandai sesi masuk pengguna Anda selesai.
3. Di aplikasi Anda, catat nilai yang ditampilkan Amazon Cognito `NewDeviceMetadata` dalam `InitiateAuth` respons `RespondToAuthChallenge` atau responsnya: `DeviceGroupKey` dan `DeviceKey`
4. Hasilkan rahasia SRP baru untuk pengguna Anda: garam dan verifikasi kata sandi. Fungsi ini tersedia di SDK yang menyediakan pustaka SRP.
5. Minta pengguna Anda untuk nama perangkat, atau buat nama dari karakteristik perangkat pengguna Anda.
6. Berikan token akses, kunci perangkat, nama perangkat, dan rahasia SRP pengguna Anda dalam permintaan [ConfirmDevice](#) API. Jika kumpulan pengguna Anda disetel ke Selalu ingat perangkat, pendaftaran pengguna Anda selesai.
7. Jika Amazon Cognito merespons `ConfirmDevice` dengan `"UserConfirmationNecessary": true`, minta pengguna Anda untuk memilih apakah mereka ingin mengingat perangkat tersebut. Jika mereka menegaskan bahwa mereka ingin mengingat perangkat, buat permintaan [UpdateDeviceStatus](#) API dengan token akses pengguna Anda, kunci perangkat, dan `"DeviceRememberedStatus": "remembered"`.
8. Jika Anda telah menginstruksikan Amazon Cognito untuk mengingat perangkat, saat berikutnya mereka masuk, alih-alih tantangan MFA, mereka akan disajikan dengan tantangan `DEVICE_SRP_AUTH`



## Masuk dengan perangkat

Setelah Anda mengonfigurasi perangkat pengguna agar diingat, Amazon Cognito tidak lagi mengharuskan mereka untuk mengirimkan kode MFA saat mereka masuk dengan kunci perangkat yang sama. Otentikasi perangkat hanya menggantikan tantangan autentikasi MFA dengan tantangan otentikasi perangkat. Anda tidak dapat memasukkan pengguna hanya dengan autentikasi perangkat. Pengguna Anda harus terlebih dahulu menyelesaikan otentikasi dengan kata sandi mereka atau tantangan khusus. Berikut ini adalah proses otentikasi untuk pengguna pada perangkat yang diingat.

Untuk melakukan autentikasi perangkat dalam alur yang menggunakan [pemicu Lambda tantangan autentikasi kustom](#), teruskan `DEVICE_KEY` parameter dalam permintaan API Anda. [InitiateAuth](#) Setelah pengguna Anda berhasil menyelesaikan semua tantangan dan `CUSTOM_CHALLENGE` tantangan mengembalikan `issueTokens` nilai `true`, Amazon Cognito mengembalikan satu `DEVICE_SRP_AUTH` tantangan terakhir.

Untuk masuk dengan perangkat

1. Ambil kunci perangkat pengguna Anda dari penyimpanan klien.
2. Mulai sesi login pengguna Anda dengan permintaan [InitiateAuth](#) API. Pilih `AuthFlow` dari `USER_SRP_AUTH`, `REFRESH_TOKEN_AUTH`, `USER_PASSWORD_AUTH`, atau `CUSTOM_AUTH`. Di `AuthParameters`, tambahkan kunci perangkat pengguna Anda ke `DEVICE_KEY` parameter, dan sertakan parameter lain yang diperlukan untuk alur masuk yang Anda pilih.
  - a. Anda juga dapat meneruskan `DEVICE_KEY` parameter `PASSWORD_VERIFIER` respons terhadap tantangan otentikasi.
3. Selesaikan respons tantangan sampai Anda menerima `DEVICE_SRP_AUTH` tantangan dalam respons.
4. Dalam permintaan [RespondToAuthChallenge](#) API, kirim `ChallengeName` dari `DEVICE_SRP_AUTH` dan parameter untuk `USERNAME`, `DEVICE_KEY`, dan `SRP_A`.
5. Amazon Cognito merespons dengan sebuah tantangan. `DEVICE_PASSWORD_VERIFIER` Respons tantangan ini mencakup nilai-nilai untuk `SECRET_BLOCK` dan `SRP_B`.
6. Dengan pustaka SRP Anda, buat dan kirimkan `PASSWORD_CLAIM_SIGNATURE`, `PASSWORD_CLAIM_SECRET_BLOCK`, `TIMESTAMP`, `USERNAME`, dan `DEVICE_KEY` parameter. Kirimkan ini dalam `RespondToAuthChallenge` permintaan tambahan.
7. Selesaikan tantangan tambahan hingga Anda menerima JWT pengguna.

Pseudocode berikut menunjukkan cara menghitung nilai untuk respons tantangan Anda `DEVICE_PASSWORD_VERIFIER`.

```
PASSWORD_CLAIM_SECRET_BLOCK = SECRET_BLOCK
TIMESTAMP = Tue Sep 25 00:09:40 UTC 2018
PASSWORD_CLAIM_SIGNATURE = Base64(SHA256_HMAC(K_USER, DeviceGroupKey + DeviceKey +
  PASSWORD_CLAIM_SECRET_BLOCK + TIMESTAMP))
K_USER = SHA256_HASH(S_USER)
S_USER = (SRP_B - k * gx)(a + ux)
x = SHA256_HASH(salt + FULL_PASSWORD)
u = SHA256_HASH(SRP_A + SRP_B)
k = SHA256_HASH(N + g)
```

## Melihat, memperbarui, dan melupakan perangkat

Anda dapat menerapkan fitur berikut di aplikasi Anda dengan Amazon Cognito API.

1. Menampilkan informasi tentang perangkat pengguna saat ini.
2. Menampilkan daftar semua perangkat pengguna Anda.
3. Lupakan perangkat.
4. Perbarui status perangkat yang diingat.

Token akses yang mengotorisasi permintaan API dalam deskripsi berikut harus menyertakan ruang lingkup `aws.cognito.signin.user.admin`. Amazon Cognito menambahkan klaim untuk cakupan ini ke semua token akses yang Anda hasilkan dengan API kumpulan pengguna Amazon Cognito. Pihak ketiga IdPs harus mengelola perangkat dan MFA secara terpisah untuk penggunanya yang mengautentikasi ke Amazon Cognito. Di UI yang dihosting, Anda dapat meminta `aws.cognito.signin.user.admin` ruang lingkup, tetapi UI yang dihosting secara otomatis menambahkan informasi perangkat ke log pengguna keamanan tingkat lanjut, dan tidak menawarkan untuk mengingat perangkat.

### Menampilkan informasi tentang perangkat

Anda dapat menanyakan informasi tentang perangkat pengguna untuk menentukan apakah masih digunakan saat ini. Misalnya, Anda mungkin ingin menonaktifkan perangkat yang diingat setelah perangkat tersebut tidak masuk selama 90 hari.

- Untuk menampilkan informasi perangkat pengguna Anda di aplikasi klien-publik, kirimkan kunci akses dan kunci perangkat pengguna Anda dalam permintaan [GetDeviceAPI](#).

- Untuk menampilkan informasi perangkat pengguna Anda di aplikasi klien rahasia, tandatangani permintaan [AdminGetDevice](#) API dengan AWS kredensi dan kirimkan nama pengguna, kunci perangkat, dan kumpulan pengguna Anda.

### Menampilkan daftar semua perangkat pengguna Anda

Anda dapat menampilkan daftar semua perangkat pengguna Anda dan propertinya. Misalnya, Anda mungkin ingin memverifikasi bahwa perangkat saat ini cocok dengan perangkat yang diingat.

- Di aplikasi klien-publik, kirimkan token akses pengguna Anda dalam permintaan [ListDevices](#) API.
- Di aplikasi klien rahasia, tandatangani permintaan [AdminListDevices](#) API dengan AWS kredensial dan kirimkan nama pengguna dan kumpulan pengguna Anda.

### Lupakan perangkat

Anda dapat menghapus kunci perangkat pengguna. Anda mungkin ingin melakukan ini ketika Anda menentukan bahwa pengguna Anda tidak lagi menggunakan perangkat, atau ketika Anda mendeteksi aktivitas yang tidak biasa dan ingin meminta pengguna untuk menyelesaikan MFA lagi. Untuk mendaftarkan perangkat lagi nanti, Anda harus membuat dan menyimpan kunci perangkat baru.

- Di aplikasi klien-publik, kirimkan kunci perangkat dan token akses pengguna Anda dalam permintaan [ForgetDevice](#) API.
- Di aplikasi klien rahasia, kirimkan kunci perangkat dan token akses pengguna Anda dalam [AdminForgetDevice](#) permintaan API.


## Menggunakan API kumpulan pengguna Amazon Cognito dan titik akhir kumpulan pengguna

Saat Anda ingin mendaftar, masuk, dan mengelola pengguna di kumpulan pengguna, Anda memiliki dua opsi.

1. Titik akhir kumpulan pengguna Anda mencakup [UI yang dihosting](#) dan titik [akhir federasi](#). Mereka membuat paket halaman web publik yang diaktifkan Amazon Cognito saat [Anda memilih domain untuk kumpulan pengguna Anda](#). Untuk memulai dengan cepat dengan fitur otentikasi dan otorisasi kumpulan pengguna Amazon Cognito, termasuk halaman untuk pendaftaran, masuk,

manajemen kata sandi, dan otentikasi multi-faktor (MFA), gunakan antarmuka pengguna bawaan dari UI yang dihosting. Titik akhir kumpulan pengguna lainnya memfasilitasi otentikasi dengan penyedia identitas pihak ketiga (IdPs). Layanan yang mereka lakukan meliputi yang berikut ini.

- a. Titik akhir panggilan balik penyedia layanan untuk klaim yang diautentikasi dari Anda, `saml2/idpresponse` dan `oauth2/idpresponse`. Jika Amazon Cognito adalah penyedia layanan perantara (SP) antara aplikasi Anda dan idP Anda, titik akhir callback mewakili layanan.
  - b. Endpoint yang memberikan informasi tentang lingkungan Anda, seperti `oauth2/userInfo` dan `jwtks.json`. Aplikasi Anda menggunakan titik akhir ini saat memverifikasi token atau mengambil data profil pengguna dengan AWS SDK dan pustaka OAuth 2.0.
2. [API kumpulan pengguna Amazon Cognito](#) adalah seperangkat alat untuk web atau aplikasi seluler Anda, setelah mengumpulkan informasi masuk di front end kustom Anda sendiri, untuk mengautentikasi pengguna. Autentikasi API kumpulan pengguna menghasilkan token web JSON berikut.
- a. Token identitas dengan klaim atribut yang dapat diverifikasi dari pengguna Anda.
  - b. [Token akses yang memberi wewenang kepada pengguna Anda untuk membuat permintaan API yang diotorisasi token ke titik akhir layanan. AWS](#)

 Note

Secara default, token akses dari kumpulan pengguna otentikasi API hanya berisi `aws.cognito.signin.user.admin` ruang lingkup. Untuk menghasilkan token akses dengan cakupan tambahan, misalnya untuk mengotorisasi permintaan ke API pihak ketiga, minta cakupan selama autentikasi melalui titik akhir kumpulan pengguna Anda atau tambahkan cakupan khusus di file. [Pemicu Lambda generasi pra token](#) Kustomisasi token akses menambah biaya ke AWS tagihan Anda.

Anda dapat menautkan pengguna federasi, yang biasanya masuk melalui titik akhir kumpulan pengguna, dengan pengguna yang profilnya lokal ke kumpulan pengguna Anda. Pengguna lokal ada secara eksklusif di direktori kumpulan pengguna Anda tanpa federasi melalui iDP eksternal. Jika Anda menautkan identitas federasi mereka ke pengguna lokal dalam permintaan [AdminLinkProviderForUserAPI](#), mereka dapat masuk dengan API kumpulan pengguna. Untuk informasi selengkapnya, lihat [Menautkan pengguna gabungan ke profil pengguna yang ada](#).

API kumpulan pengguna Amazon Cognito memiliki tujuan ganda. Ini membuat dan mengonfigurasi sumber daya kumpulan pengguna Amazon Cognito Anda. Misalnya, Anda dapat membuat kumpulan pengguna, menambahkan AWS Lambda pemicu, dan mengonfigurasi domain UI yang dihosting. API kumpulan pengguna juga melakukan pendaftaran, masuk, dan operasi pengguna lainnya untuk pengguna lokal dan tertaut.

Contoh skenario dengan API kumpulan pengguna Amazon Cognito

1. Pengguna Anda memilih tombol “Buat akun” yang Anda buat di aplikasi Anda. Mereka memasukkan alamat email dan kata sandi.
2. Aplikasi Anda mengirimkan permintaan [SignUp](#) API dan membuat pengguna baru di kumpulan pengguna Anda.
3. Aplikasi Anda meminta pengguna Anda untuk kode konfirmasi email. Pengguna Anda memasukkan kode yang mereka terima dalam pesan email.
4. Aplikasi Anda mengirimkan permintaan [ConfirmSignUp](#) API dengan kode konfirmasi pengguna.
5. Aplikasi Anda meminta pengguna Anda untuk nama pengguna dan kata sandi mereka, dan mereka memasukkan informasi mereka.
6. Aplikasi Anda mengirimkan permintaan [InitiateAuth](#) API dan menyimpan token ID, token akses, dan token penyegaran. Aplikasi Anda memanggil library OIDC untuk mengelola token pengguna Anda dan mempertahankan sesi persisten untuk pengguna tersebut.

Di API kumpulan pengguna Amazon Cognito, Anda tidak dapat memasukkan pengguna yang melakukan federasi melalui iDP. Anda harus mengautentikasi pengguna ini melalui titik akhir kumpulan pengguna Anda. Untuk informasi selengkapnya tentang titik akhir kumpulan pengguna yang menyertakan UI yang dihosting, lihat [Titik akhir federasi kumpulan pengguna dan referensi UI yang dihosting](#). Pengguna federasi Anda dapat memulai di UI yang dihosting dan memilih IDP mereka, atau Anda dapat melewati UI yang dihosting dan mengirim pengguna Anda langsung ke IDP Anda untuk masuk. Ketika permintaan API Anda ke [Otorisasi titik akhir](#) menyertakan parameter iDP, Amazon Cognito secara diam-diam mengalihkan pengguna Anda ke halaman login iDP.

Contoh skenario dengan titik akhir kumpulan pengguna

1. Pengguna Anda memilih tombol “Buat akun” yang Anda buat di aplikasi Anda.
2. Anda menyajikan pengguna Anda dengan daftar penyedia identitas sosial di mana Anda telah mendaftarkan kredensi pengembang. Pengguna Anda memilih Apple.

3. Aplikasi Anda memulai permintaan ke nama `SignInWithApple` penyedia [Otorisasi titik akhir with](#).
4. Browser pengguna Anda membuka halaman otorisasi Apple OAuth. Pengguna Anda memilih untuk mengizinkan Amazon Cognito membaca informasi profil mereka.
5. Amazon Cognito mengonfirmasi token akses Apple dan menanyakan profil Apple pengguna Anda.
6. Pengguna Anda menyajikan kode otorisasi Amazon Cognito ke aplikasi Anda.
7. Aplikasi Anda menukar kode otorisasi dengan [Titik akhir token](#) dan menyimpan token ID, token akses, dan token penyegaran. Aplikasi Anda memanggil library OIDC untuk mengelola token pengguna Anda dan mempertahankan sesi persisten untuk pengguna tersebut.

API kumpulan pengguna dan titik akhir kumpulan pengguna mendukung berbagai skenario, yang dijelaskan di seluruh panduan ini. Bagian berikut memeriksa bagaimana API kumpulan pengguna membagi lebih lanjut ke dalam kelas-kelas yang mendukung persyaratan pendaftaran, masuk, dan pengelolaan sumber daya Anda.

## Kumpulan pengguna Amazon Cognito operasi API yang diautentikasi dan tidak diautentikasi

API kumpulan pengguna Amazon Cognito, baik antarmuka manajemen sumber daya dan antarmuka otentikasi dan otorisasi yang dihadapi pengguna, menggabungkan model otorisasi yang mengikuti operasinya. Bergantung pada operasi API, Anda mungkin harus memberikan otorisasi dengan kredensial IAM, token akses, token sesi, rahasia klien, atau kombinasi keduanya. Untuk banyak operasi otentikasi dan otorisasi pengguna, Anda memiliki pilihan versi permintaan yang diautentikasi dan tidak diautentikasi. Operasi yang tidak diautentikasi adalah praktik keamanan terbaik untuk aplikasi yang Anda distribusikan kepada pengguna, seperti aplikasi seluler; Anda tidak perlu menyertakan rahasia apa pun dalam kode Anda.

Anda hanya dapat menetapkan izin dalam kebijakan IAM untuk dan. [Operasi manajemen yang diautentikasi oleh IAM](#) [Operasi pengguna yang diautentikasi oleh IAM](#)

### Operasi manajemen yang diautentikasi oleh IAM

Operasi manajemen yang diautentikasi oleh IAM memodifikasi dan melihat kumpulan pengguna dan konfigurasi klien aplikasi Anda, seperti yang akan Anda lakukan di. AWS Management Console

Misalnya, untuk mengubah kumpulan pengguna Anda dalam permintaan [UpdateUserPoolAPI](#), Anda harus menunjukkan AWS kredensi dan izin IAM untuk memperbarui sumber daya.

Untuk mengotorisasi permintaan ini di AWS Command Line Interface (AWS CLI) atau AWS SDK, konfigurasi lingkungan Anda dengan variabel lingkungan atau konfigurasi klien yang menambahkan kredensial IAM ke permintaan Anda. Untuk informasi selengkapnya, lihat [Mengakses AWS menggunakan AWS kredensial Anda](#) di. Referensi Umum AWS Anda juga dapat mengirim permintaan langsung ke [titik akhir layanan untuk API](#) kumpulan pengguna Amazon Cognito. Anda harus mengotorisasi, atau menandatangani, permintaan ini dengan AWS kredensial yang Anda sematkan di header permintaan Anda. Untuk informasi selengkapnya, lihat [Menandatangani permintaan AWS API](#).

Operasi manajemen yang diautentikasi oleh IAM

AddCustomAttributes

CreateGroup

CreateIdentityProvider

CreateResourceServer

CreateUserImportJob

CreateUserPool

CreateUserPoolClient

CreateUserPoolDomain

DeleteGroup

DeleteIdentityProvider

DeleteResourceServer

DeleteUserPool

DeleteUserPoolClient

DeleteUserPoolDomain

Operasi manajemen yang diautentikasi oleh IAM

DescribeIdentityProvider

DescribeResourceServer

DescribeRiskConfiguration

DescribeUserImportJob

DescribeUserPool

DescribeUserPoolClient

DescribeUserPoolDomain

GetCSVHeader

GetGroup

GetIdentityProviderByIdentifier

GetSigningCertificate

GetUICustomization

GetUserPoolMfaConfig

ListGroups

ListIdentityProviders

ListResourceServers

ListTagsForResource

ListUserImportJobs

ListUserPoolClients

ListUserPools

ListUsers



Operasi manajemen yang diautentikasi oleh IAM

ListUsersInGroup

SetRiskConfiguration

SetUICustomization

SetUserPoolMfaConfig

StartUserImportJob

StopUserImportJob

TagResource

UntagResource

UpdateGroup

UpdateIdentityProvider

UpdateResourceServer

UpdateUserPool

UpdateUserPoolClient

UpdateUserPoolDomain

Operasi pengguna yang diautentikasi oleh IAM

Operasi pengguna yang diautentikasi IAM mendaftar, masuk, mengelola kredensi untuk, memodifikasi, dan melihat pengguna Anda.

Misalnya, Anda dapat memiliki tingkat aplikasi sisi server yang mendukung ujung depan web. Aplikasi sisi server Anda adalah klien rahasia OAuth yang Anda percayai dengan akses istimewa ke sumber daya Amazon Cognito Anda. Untuk mendaftarkan pengguna di aplikasi, server Anda dapat menyertakan AWS kredensial dalam permintaan [AdminCreateUserAPI](#). Untuk informasi selengkapnya tentang jenis klien OAuth, lihat [Jenis Klien](#) di Kerangka Otorisasi OAuth 2.0.

Untuk mengotorisasi permintaan ini di AWS CLI atau AWS SDK, konfigurasi lingkungan aplikasi sisi server Anda dengan variabel lingkungan atau konfigurasi klien yang menambahkan kredensial IAM ke permintaan Anda. Untuk informasi selengkapnya, lihat [Mengakses AWS menggunakan AWS kredensial Anda](#) di Referensi Umum AWS Anda juga dapat mengirim permintaan langsung ke [titik akhir layanan untuk API](#) kumpulan pengguna Amazon Cognito. Anda harus mengotorisasi, atau menandatangani, permintaan ini dengan AWS kredensial yang Anda sematkan di header permintaan Anda. Untuk informasi selengkapnya, lihat [Menandatangani permintaan AWS API](#).

Jika klien aplikasi Anda memiliki rahasia klien, Anda harus memberikan kredensi IAM dan, tergantung pada operasinya, SecretHash parameter atau nilai dalam SECRET\_HASH. AuthParameters Untuk informasi selengkapnya, lihat [Menghitung nilai hash rahasia](#).

Operasi pengguna yang diautentikasi oleh IAM

AdminAddUserToGroup

AdminConfirmSignUp

AdminCreateUser

AdminDeleteUser

AdminDeleteUserAttributes

AdminDisableProviderForUser

AdminDisableUser

AdminEnableUser

AdminForgetDevice

AdminGetDevice

AdminGetUser

AdminInitiateAuth

AdminLinkProviderForUser

AdminListDevices

Operasi pengguna yang diautentikasi oleh IAM

`AdminListGroupsWithUser`

`AdminListUserAuthEvents`

`AdminRemoveUserFromGroup`

`AdminResetUserPassword`

`AdminRespondToAuthChallenge`

`AdminSetUserMFAResponse`

`AdminSetUserPassword`

`AdminSetUserSettings`

`AdminUpdateAuthEventFeedback`

`AdminUpdateDeviceStatus`

`AdminUpdateUserAttributes`

`AdminUserGlobalSignOut`

Operasi pengguna yang tidak diautentikasi

Operasi pengguna yang tidak diautentikasi mendaftar, masuk, dan memulai pengaturan ulang kata sandi untuk pengguna Anda. Gunakan operasi API yang tidak diautentikasi, atau publik, saat Anda ingin siapa pun di internet mendaftar dan masuk ke aplikasi Anda.

Misalnya, untuk mendaftarkan pengguna di aplikasi, Anda dapat mendistribusikan klien publik OAuth yang tidak memberikan akses istimewa apa pun ke rahasia. Anda dapat mendaftarkan pengguna ini dengan operasi API yang tidak diautentikasi. [SignUp](#)

Untuk mengirim permintaan ini di klien publik yang Anda kembangkan dengan AWS SDK, Anda tidak perlu mengonfigurasi kredensi apa pun. Anda juga dapat mengirim permintaan langsung ke [titik akhir layanan](#) untuk API kumpulan pengguna Amazon Cognito tanpa otorisasi tambahan.

Jika klien aplikasi Anda memiliki rahasia klien, Anda harus memberikan, tergantung pada operasinya, `SecretHash` parameter atau `SECRET_HASH` nilai dalam `AuthParameters`. Untuk informasi selengkapnya, lihat [Menghitung nilai hash rahasia](#).

Operasi pengguna yang tidak diautentikasi

`SignUp`

`ConfirmSignUp`

`ResendConfirmationCode`

`ForgotPassword`

`ConfirmForgotPassword`

`InitiateAuth`

Operasi pengguna yang diotorisasi token

Operasi pengguna yang diotorisasi token keluar, mengelola kredensial untuk, memodifikasi, dan melihat pengguna Anda setelah mereka masuk atau memulai proses masuk. Gunakan operasi API yang diotorisasi token jika Anda tidak ingin mendistribusikan rahasia di aplikasi, dan Anda ingin mengotorisasi permintaan dengan kredensial pengguna Anda sendiri. Jika pengguna Anda telah menyelesaikan proses masuk, Anda harus mengotorisasi permintaan API resmi token mereka dengan token akses. Jika pengguna Anda berada di tengah proses masuk, Anda harus mengotorisasi permintaan API resmi token mereka dengan token sesi yang dikembalikan Amazon Cognito sebagai respons terhadap permintaan sebelumnya.

Misalnya, di klien publik, Anda mungkin ingin memperbarui profil pengguna dengan cara yang membatasi akses tulis ke profil pengguna sendiri saja. Untuk melakukan pembaruan ini, klien Anda dapat menyertakan token akses pengguna dalam permintaan [UpdateUserAttributesAPI](#).

Untuk mengirim permintaan ini di klien publik yang Anda kembangkan dengan AWS SDK, Anda tidak perlu mengonfigurasi kredensi apa pun. Sertakan `Session` parameter `AccessToken` atau dalam permintaan Anda. Anda juga dapat mengirim permintaan langsung ke [titik akhir layanan untuk API](#) kumpulan pengguna Amazon Cognito. Untuk mengotorisasi permintaan ke titik akhir layanan, sertakan token akses atau sesi di badan POST permintaan Anda.

Untuk menandatangani permintaan API untuk operasi yang diotorisasi token, sertakan token akses sebagai Authorization header dalam permintaan Anda, dalam format. Bearer *<Base64-encoded access token>*

Operasi pengguna yang diotorisasi token	AccessToken	Sesi
RespondToAuthChallenge		✓
ChangePassword	✓	
GetUser	✓	
UpdateUserAttributes	✓	
DeleteUserAttributes	✓	
DeleteUser	✓	
ConfirmDevice	✓	
ForgetDevice	✓	
GetDevice	✓	
ListDevices	✓	
UpdateDeviceStatus	✓	
GetUserAttributeVerificationCode	✓	
VerifyUserAttribute	✓	

Operasi pengguna yang diotorisasi token	AccessTok en	Sesi
SetUserSettings	✓	
SetUserMF AReferensi	✓	
GlobalSignOut	✓	
Associate SoftwareToken	✓	✓
UpdateAuthEventFeedback		✓
VerifySoftwareToken	✓	✓
RevokeToken <sup>1</sup>		

<sup>1</sup> RevokeToken mengambil token penyegaran sebagai parameter. Token penyegaran berfungsi sebagai token otorisasi, dan sebagai sumber daya target.

## Memperbarui konfigurasi kumpulan pengguna

Untuk mengubah pengaturan kumpulan pengguna Amazon Cognito di AWS Management Console, navigasikan tab berbasis fitur di pengaturan kumpulan pengguna dan perbarui bidang seperti yang dijelaskan di area lain dalam panduan ini. Anda tidak dapat mengubah beberapa pengaturan setelah membuat kumpulan pengguna. Jika Anda ingin mengubah pengaturan berikut, Anda harus membuat kumpulan pengguna atau klien aplikasi baru.

### Nama kolam pengguna

Nama parameter API: [PoolName](#)

Nama ramah yang Anda tetapkan ke kumpulan pengguna Anda. Untuk mengubah nama kumpulan pengguna, buat kumpulan pengguna baru.

## Opsi masuk kumpulan pengguna Amazon Cognito

Nama parameter API: [AliasAttributes](#) dan [UsernameAttributes](#)

Atribut yang dapat diteruskan pengguna Anda sebagai nama pengguna saat mereka masuk. Saat membuat kumpulan pengguna, Anda dapat memilih untuk mengizinkan masuk dengan nama pengguna, alamat email, nomor telepon, atau nama pengguna pilihan. Untuk mengubah opsi masuk kumpulan pengguna, buat kumpulan pengguna baru.

### Jadikan nama pengguna sensitif huruf besar/huruf besar

Nama parameter API: [UsernameConfiguration](#)

Saat Anda membuat nama pengguna yang cocok dengan nama pengguna lain kecuali untuk kasus huruf, Amazon Cognito dapat memperlakukannya sebagai pengguna yang sama atau sebagai pengguna unik. Untuk informasi selengkapnya, lihat [Sensitivitas casing kumpulan pengguna](#). Untuk mengubah sensitivitas huruf besar, buat kumpulan pengguna baru.

### Rahasia klien

Nama parameter API: [GenerateSecret](#)

Saat membuat klien aplikasi, Anda dapat membuat rahasia klien sehingga hanya sumber tepercaya yang dapat mengajukan permintaan ke kumpulan pengguna Anda. Untuk informasi selengkapnya, lihat [Klien aplikasi kumpulan pengguna](#). Untuk mengubah rahasia klien, buat klien aplikasi baru di kumpulan pengguna yang sama.

### Atribut yang diperlukan

Nama parameter API: [Skema](#)

Atribut yang harus diberikan nilai pengguna saat mereka mendaftar, atau saat Anda membuatnya. Untuk informasi selengkapnya, lihat [Atribut kumpulan pengguna](#). Untuk mengubah atribut yang diperlukan, buat kumpulan pengguna baru.

### Atribut kustom

Nama parameter API: [Skema](#)

Atribut dengan nama khusus. Anda dapat mengubah nilai atribut kustom pengguna, tetapi Anda tidak dapat menghapus atribut kustom dari kumpulan pengguna Anda. Untuk informasi selengkapnya, lihat [Atribut kumpulan pengguna](#). Jika Anda mencapai jumlah maksimum atribut kustom dan Anda ingin mengubah daftar, buat kumpulan pengguna baru.

## Konfigurasi SMS

Setelah Anda mengaktifkan pesan SMS di kumpulan pengguna Anda, Anda tidak dapat menonaktifkannya.

- Jika Anda memilih untuk mengonfigurasi pesan SMS saat membuat kumpulan pengguna, Anda tidak dapat menonaktifkan SMS setelah menyelesaikan penyiapan.
- Anda dapat mengaktifkan pesan SMS di kumpulan pengguna yang Anda buat, tetapi setelah itu Anda tidak dapat menonaktifkan SMS.
- Amazon Cognito dapat menggunakan pesan SMS untuk undangan dan pemulihan akun pengguna, verifikasi atribut, dan otentikasi multi-faktor (MFA). Setelah Anda mengaktifkan pesan SMS, Anda dapat mengaktifkan atau menonaktifkan pesan SMS untuk fungsi-fungsi ini kapan saja.
- Konfigurasi pesan SMS menyertakan peran IAM yang Anda delegasikan ke Amazon Cognito untuk mengirim pesan dengan Amazon SNS. Anda dapat mengubah peran yang ditetapkan kapan saja.

## Memperbarui kumpulan pengguna dengan AWS SDK, AWS CDK, atau REST API

Di konsol Amazon Cognito, Anda dapat mengubah pengaturan kumpulan pengguna satu parameter sekaligus. Misalnya, untuk menambahkan pemicu Lambda, Anda memilih Add Lambda trigger dan memilih fungsi dan tipe pemicu. API kumpulan pengguna Amazon Cognito disusun sedemikian rupa sehingga operasi pembaruan untuk kumpulan pengguna dan klien aplikasi memerlukan set parameter lengkap untuk kumpulan pengguna. Namun, konsol secara transparan mengotomatiskan operasi pembaruan ini dengan pengaturan kumpulan pengguna Anda yang lain.

Terkadang Anda mungkin menemukan bahwa perubahan di tempat lain Akun AWS dapat menyebabkan pembaruan menghasilkan kesalahan saat tidak terkait dengan pengaturan yang ingin Anda ubah. Identitas Amazon SES yang dihapus atau perubahan izin IAM untuk AWS WAF, misalnya. Jika salah satu parameter saat ini tidak lagi valid, Anda tidak dapat memperbarui pengaturan hingga Anda memperbaikinya. Ketika Anda menemukan kesalahan seperti itu, periksa respons kesalahan dan validasi pengaturan yang disebutkan.

[Kumpulan pengguna Amazon Cognito REST API](#) dan [AWS SDK](#) adalah alat untuk otomatisasi dan konfigurasi terprogram sumber daya Amazon Cognito. [AWS Cloud Development Kit \(AWS CDK\)](#) Permintaan dengan alat ini juga harus, seperti konsol Amazon Cognito, memperbarui pengaturan dengan konfigurasi sumber daya lengkap di badan permintaan. Pada tingkat tinggi, Anda harus melakukan proses berikut.



1. Tangkap output dari operasi yang menjelaskan konfigurasi sumber daya yang ada.
2. Ubah output dengan perubahan pengaturan Anda.
3. Kirim konfigurasi yang dimodifikasi dalam operasi yang memperbarui sumber daya Anda.

Prosedur berikut memperbarui konfigurasi Anda dengan operasi [UpdateUserPoolAPI](#). Pendekatan yang sama, dengan bidang input yang berbeda, berlaku untuk [UpdateUserPoolClient](#).

#### Important

Jika Anda tidak memberikan nilai untuk parameter yang ada, Amazon Cognito menyetelnya ke nilai default. Misalnya, ketika Anda sudah ada LambdaConfig dan Anda mengirimkan UpdateUserPool dengan kosongLambdaConfig, Anda menghapus penetapan semua fungsi Lambda ke pemicu kumpulan pengguna. Rencanakan sesuai saat Anda ingin mengotomatiskan perubahan pada konfigurasi kumpulan pengguna Anda.

1. Tangkap status kumpulan pengguna Anda yang ada dengan [DescribeUserPool](#).
2. Format output DescribeUserPool untuk mencocokkan [parameter permintaan](#) UpdateUserPool. Hapus bidang tingkat atas berikut dan objek anak mereka dari output JSON.
  - Arn
  - CreationDate
  - CustomDomain
    - Perbarui bidang ini dengan operasi [UpdateUserPoolDomainAPI](#).
  - Domain
    - Perbarui bidang ini dengan operasi [UpdateUserPoolDomainAPI](#).
  - EmailConfigurationFailure
  - EstimatedNumberOfUsers
  - Id
  - LastModifiedDate
  - Name
  - SchemaAttributes
  - SmsConfigurationFailure

- Status
3. Konfirmasikan bahwa JSON yang dihasilkan cocok dengan [parameter permintaan UpdateUserPool](#).
  4. Ubah parameter apa pun yang ingin Anda ubah di JSON yang dihasilkan.
  5. Kirim permintaan UpdateUserPool API dengan JSON Anda yang dimodifikasi sebagai input permintaan.

Anda juga dapat menggunakan DescribeUserPool output yang dimodifikasi ini update-user-pool dalam --cli-input-json parameter di AWS CLI.

Bergantian, jalankan AWS CLI perintah berikut untuk menghasilkan JSON dengan nilai kosong untuk bidang input yang diterima untuk update-user-pool Anda kemudian dapat mengisi bidang ini dengan nilai yang ada dari kumpulan pengguna Anda.

```
aws cognito-idp update-user-pool --generate-cli-skeleton --output json
```

Jalankan perintah berikut untuk menghasilkan objek JSON yang sama untuk klien aplikasi.

```
aws cognito-idp update-user-pool-client --generate-cli-skeleton --output json
```

## Menyiapkan dan menggunakan UI yang dihosting Amazon Cognito dan titik akhir federasi

Kumpulan pengguna Amazon Cognito dengan domain adalah server otorisasi yang sesuai dengan OAuth-2.0 dan antarmuka pengguna (UI) yang dihosting untuk otentikasi. ready-to-use Server otorisasi merutekan permintaan otentikasi, mengeluarkan dan mengelola token web JSON (JWT), dan memberikan informasi atribut pengguna. UI yang dihosting adalah kumpulan antarmuka web untuk pendaftaran dasar, masuk, otentikasi multi-faktor, dan aktivitas pengaturan ulang kata sandi di kumpulan pengguna Anda. Ini juga merupakan hub pusat untuk otentikasi dengan penyedia identitas pihak ketiga (IdPs) yang Anda kaitkan dengan aplikasi Anda. Aplikasi Anda dapat memanggil UI yang dihosting dan titik akhir otorisasi saat Anda ingin mengotentikasi dan mengotorisasi pengguna. Anda dapat membuat pengalaman pengguna UI yang dihosting sesuai dengan merek Anda dengan logo dan kustomisasi CSS Anda sendiri. Untuk informasi selengkapnya tentang komponen UI yang dihosting dan server otorisasi, lihat [Titik akhir federasi kumpulan pengguna dan referensi UI yang dihosting](#).

**Note**

UI yang dihosting Amazon Cognito tidak mendukung otentikasi khusus dengan [tantangan otentikasi khusus](#) yang memicu Lambda.

## Topik

- [Menyiapkan UI yang dihosting dengan AWS Amplify](#)
- [Menyiapkan UI yang dihosting dengan konsol Amazon Cognito](#)
- [Melihat halaman masuk Anda](#)
- [Hal-hal yang perlu diketahui tentang UI yang dihosting kumpulan pengguna Amazon Cognito](#)
- [Mengkonfigurasi domain kolam pengguna](#)
- [Menyesuaikan halaman web masuk dan daftar bawaan](#)
- [Mendaftar dan masuk dengan UI yang dihosting](#)

## Menyiapkan UI yang dihosting dengan AWS Amplify

Jika Anda menggunakan AWS Amplify untuk menambahkan otentikasi ke web atau aplikasi seluler, Anda dapat mengatur UI yang dihosting dengan menggunakan antarmuka baris perintah (CLI) dan pustaka dalam kerangka kerja. AWS Amplify Untuk menambahkan autentikasi ke aplikasi Anda, Anda menggunakan AWS Amplify CLI untuk menambahkan kategori Auth ke project Anda. Kemudian, dalam kode klien Anda, Anda menggunakan AWS Amplify pustaka untuk mengautentikasi pengguna dengan kumpulan pengguna Amazon Cognito Anda.

Anda dapat menampilkan UI yang dihosting sebelumnya, atau Anda dapat menggabungkan pengguna melalui titik akhir OAuth 2.0 yang mengalihkan ke penyedia masuk sosial, seperti Facebook, Google, Amazon, atau Apple. Setelah pengguna berhasil melakukan autentikasi dengan penyedia sosial, AWS Amplify buat pengguna baru di kumpulan pengguna Anda jika diperlukan, lalu berikan token OIDC pengguna ke aplikasi Anda.

Contoh berikut menunjukkan cara menggunakan AWS Amplify untuk menyiapkan UI yang dihosting dengan penyedia sosial di aplikasi Anda.

- [AWS Amplify otentikasi untuk JavaScript.](#)
- [AWS Amplify otentikasi untuk Swift.](#)

- [AWS Amplify otentikasi untuk Flutter.](#)
- [AWS Amplify otentikasi untuk Android.](#)

## Menyiapkan UI yang dihosting dengan konsol Amazon Cognito

Buat klien aplikasi

1. Masuk ke [Konsol Amazon Cognito](#). Jika diminta, masukkan AWS kredensial Anda.
2. Pilih Kolam Pengguna.
3. Pilih kolam pengguna yang ada dari daftar, atau [buat kolam pengguna](#).
4. Pilih tab Integrasi aplikasi.
5. Di bawah Klien aplikasi, pilih Buat klien aplikasi.
6. Pilih jenis Aplikasi: Klien publik, Klien Rahasia, atau Lainnya. Klien Publik biasanya beroperasi dari perangkat pengguna Anda dan menggunakan API yang tidak diautentikasi dan diautentikasi token. Klien Rahasia biasanya beroperasi dari aplikasi di server pusat yang Anda percayai dengan rahasia klien dan kredensial API, dan menggunakan header otorisasi dan kredensial untuk menandatangani permintaan. AWS Identity and Access Management Jika kasus penggunaan Anda berbeda dari pengaturan klien aplikasi yang telah dikonfigurasi sebelumnya untuk klien Publik atau klien Rahasia, pilih Lainnya.
7. Masukkan nama klien Aplikasi.
8. Pilih alur Autentikasi yang ingin Anda izinkan di klien aplikasi Anda.
9. Konfigurasi durasi sesi alur Otentikasi. Ini adalah jumlah waktu yang dimiliki pengguna Anda untuk menyelesaikan setiap tantangan otentikasi sebelum token sesi mereka kedaluwarsa.
10. (Opsional) Konfigurasi kedaluwarsa token.
  - a. Tentukan kedaluwarsa token Refresh untuk klien aplikasi. Nilai default-nya adalah 30 hari. Anda dapat mengubahnya ke nilai berapa pun antara 1 jam dan 10 tahun.
  - b. Tentukan kedaluwarsa token Access untuk klien aplikasi. Nilai default adalah 1 jam. Anda dapat mengubahnya ke nilai berapa pun antara 5 menit dan 24 jam.
  - c. Tentukan kedaluwarsa token ID untuk klien aplikasi. Nilai default adalah 1 jam. Anda dapat mengubahnya ke nilai berapa pun antara 5 menit dan 24 jam.

**⚠ Important**

Jika Anda menggunakan UI yang dihosting dan mengonfigurasi masa pakai token kurang dari satu jam, pengguna Anda akan dapat menggunakan token berdasarkan durasi cookie sesi mereka, yang saat ini diperbaiki pada satu jam.

11. Pilih Hasilkan rahasia klien agar Amazon Cognito menghasilkan rahasia klien untuk Anda. Rahasia klien biasanya dikaitkan dengan klien rahasia.
12. Pilih apakah Anda akan Aktifkan pencabutan token untuk klien aplikasi ini. Ini akan meningkatkan ukuran token. Untuk informasi selengkapnya, lihat [Mencabut Token](#).
13. Pilih apakah Anda akan Mencegah pesan kesalahan yang mengungkapkan keberadaan pengguna untuk klien aplikasi ini. Amazon Cognito akan menanggapi permintaan masuk untuk pengguna yang tidak ada dengan pesan umum yang menyatakan bahwa nama pengguna atau kata sandi salah.
14. (Opsional) Konfigurasi izin baca dan tulis Atribut untuk klien aplikasi ini. Klien aplikasi Anda dapat memiliki izin untuk membaca dan menulis subset terbatas dari skema atribut kumpulan pengguna Anda.
15. Pilih Buat.
16. Perhatikan id Klien. Ini akan mengidentifikasi klien aplikasi dalam permintaan pendaftaran dan masuk.

### Konfigurasi app

1. Di tab Integrasi aplikasi, pilih klien aplikasi Anda di bawah Klien aplikasi. Tinjau informasi UI Hosted Anda saat ini.
2. Tambahkan URL callback di bawah URL callback yang diizinkan. URL callback adalah tempat pengguna dialihkan setelah login berhasil.
3. Tambahkan URL keluar di bawah URL keluar yang diizinkan. URL keluar adalah tempat pengguna Anda dialihkan setelah keluar.
4. Tambahkan setidaknya satu opsi yang tercantum dari daftar penyedia Identitas.
5. Di bawah jenis hibah OAuth 2.0, pilih Pemberian kode otorisasi untuk mengembalikan kode otorisasi yang kemudian ditukar dengan token kumpulan pengguna. Karena tidak pernah berinteraksi langsung ke pengguna akhir, token cenderung tidak dikompromikan. Namun, aplikasi kustom diperlukan pada backend untuk menukar kode otorisasi dengan token kolam

- pengguna. Untuk alasan keamanan, kami menyarankan Anda menggunakan alur pemberian kode otorisasi, bersama dengan [kunci Proof for code Exchange \(PKCE\)](#), untuk aplikasi seluler.
6. Di bawah jenis hibah OAuth 2.0, pilih Pemberian implisit agar token web JSON (JWT) kumpulan pengguna dikembalikan kepada Anda dari Amazon Cognito. Anda dapat menggunakan aliran ini ketika tidak ada backend yang tersedia untuk menukar kode otorisasi dengan token. Ini juga membantu untuk debugging token.
  7. Anda dapat mengaktifkan kode Otorisasi dan hibah kode Implisit, dan kemudian menggunakan setiap hibah sesuai kebutuhan. Jika kode Otorisasi atau hibah kode Implisit tidak dipilih dan klien aplikasi Anda memiliki rahasia klien, Anda dapat mengaktifkan hibah kredensial Klien. Pilih kredensial Klien hanya jika aplikasi Anda perlu meminta token akses atas namanya sendiri dan bukan atas nama pengguna.
  8. Pilih cakupan OpenID Connect yang ingin Anda otorisasi untuk klien aplikasi ini.
  9. Pilih Simpan perubahan.

### Mengkonfigurasi domain

1. Arahkan ke tab Integrasi aplikasi untuk kumpulan pengguna Anda.
2. Di samping Domain, pilih Tindakan dan pilih Buat domain khusus atau Buat domain Cognito. Jika Anda telah mengonfigurasi domain kumpulan pengguna, pilih Hapus domain Cognito atau Hapus domain kustom sebelum membuat domain kustom baru.
3. Masukkan awalan domain yang tersedia untuk digunakan dengan domain Cognito. Untuk informasi tentang cara menyiapkan domain Kustom, lihat [Menggunakan Domain Anda sendiri untuk UI yang dihosting](#)
4. Pilih Buat.

### Melihat halaman masuk Anda

Di konsol Amazon Cognito, pilih tombol Lihat UI yang Dihosting dalam konfigurasi klien aplikasi Anda, di bawah Klien aplikasi dan analitik di tab Integrasi aplikasi. Tombol ini akan membawa Anda ke halaman login di UI yang dihosting dengan parameter dasar berikut.

- Id klien aplikasi
- Permintaan hibah kode otorisasi
- Permintaan untuk semua cakupan yang telah Anda aktifkan untuk klien aplikasi saat ini
- URL callback pertama dalam daftar untuk klien aplikasi saat ini

Tombol Lihat UI yang dihosting berguna saat Anda ingin menguji fungsi dasar UI yang dihosting. Anda dapat menyesuaikan URL masuk Anda dengan parameter tambahan dan modifikasi. Dalam kebanyakan kasus, parameter yang dibuat secara otomatis dari tautan UI yang dihosting View tidak sepenuhnya sesuai dengan kebutuhan aplikasi Anda. Dalam kasus ini, Anda harus menyesuaikan URL yang dipanggil aplikasi saat masuk ke pengguna. Untuk informasi selengkapnya tentang kunci dan nilai parameter masuk, lihat [Titik akhir federasi kumpulan pengguna dan referensi UI yang dihosting](#).

Halaman web login UI yang dihosting menggunakan format URL berikut. Contoh ini meminta hibah kode otorisasi dengan `response_type=code` parameter.

```
https://<your domain>/oauth2/authorize?response_type=code&client_id=<your app client id>&redirect_uri=<your callback url>
```

Anda dapat mengambil string domain kumpulan pengguna dari tab Integrasi aplikasi. Di tab yang sama, Anda dapat mengidentifikasi ID klien aplikasi, URL callback mereka, cakupan yang diizinkan, dan konfigurasi lainnya di bawah Klien aplikasi dan analitik.

Saat Anda menavigasi ke `/oauth2/authorize` titik akhir dengan parameter kustom Anda, Amazon Cognito mengarahkan Anda ke `/oauth2/login` titik akhir atau, jika Anda memiliki `identity_provider` parameter `idp_identifier` atau, secara diam-diam mengarahkan Anda ke halaman masuk IDP Anda. Untuk contoh URL yang melewati UI yang dihosting, lihat [Inisiasi sesi SAMP di kumpulan pengguna Amazon Cognito](#).

Contoh permintaan UI yang dihosting untuk hibah implisit

Anda dapat melihat halaman web login UI yang dihosting dengan URL berikut untuk pemberian kode implisit di mana. `response_type=token` Setelah berhasil masuk, Amazon Cognito mengembalikan token kolam pengguna ke bilah alamat browser web Anda.

```
https://mydomain.us-east-1.amazoncognito.com/authorize?response_type=token&client_id=1example23456789&redirect_uri=https://mydomain.example.com
```

Identitas dan token akses muncul sebagai parameter yang ditambahkan ke URL pengalihan Anda.

Berikut ini adalah contoh respons dari permintaan hibah implisit.

```
https://mydomain.example.com/  
#id_token=eyJraaBcDeF1234567890&access_token=eyJraGhIjKlM1112131415&expires_in=3600&token_type=
```

## Hal-hal yang perlu diketahui tentang UI yang dihosting kumpulan pengguna Amazon Cognito

### UI yang dihosting dan mengonfirmasi pengguna sebagai administrator

Untuk pengguna lokal kumpulan pengguna, UI yang dihosting berfungsi paling baik saat Anda mengonfigurasi kumpulan pengguna agar Izinkan Cognito mengirim pesan secara otomatis untuk memverifikasi dan mengonfirmasi. Saat Anda mengaktifkan setelan ini, Amazon Cognito mengirimkan pesan dengan kode konfirmasi kepada pengguna yang mendaftar. Saat Anda mengonfirmasi pengguna sebagai administrator kumpulan pengguna, UI yang dihosting akan menampilkan pesan kesalahan setelah mendaftar. Dalam keadaan ini, Amazon Cognito telah membuat pengguna baru, tetapi belum dapat mengirim pesan verifikasi. Anda masih dapat mengonfirmasi pengguna sebagai administrator, tetapi mereka mungkin menghubungi meja dukungan Anda setelah mereka menemukan kesalahan. Untuk informasi selengkapnya tentang konfirmasi administratif, lihat [Memungkinkan pengguna untuk mendaftar di aplikasi Anda tetapi mengonfirmasinya sebagai administrator kumpulan pengguna](#).

### Melihat perubahan Anda ke konfigurasi UI yang dihosting

Jika perubahan pada halaman UI yang di-host tidak segera muncul, tunggu beberapa menit dan kemudian segarkan halaman.

### Menguraikan token kumpulan pengguna

Token kumpulan pengguna Amazon Cognito ditandatangani menggunakan algoritma RS256. Anda dapat memecahkan kode dan memverifikasi token kumpulan pengguna menggunakan AWS Lambda, lihat [Dekode, dan verifikasi token Amazon Cognito](#) JWT aktif. GitHub

### Versi UI dan TLS yang dihosting

UI yang dihosting memerlukan enkripsi dalam perjalanan. Domain kumpulan pengguna yang disediakan oleh Amazon Cognito memerlukan versi TLS minimum 1.2. Domain kustom mendukung tetapi tidak memerlukan TLS versi 1.2. Karena Amazon Cognito mengelola konfigurasi UI yang



dihosting dan titik akhir server otorisasi, Anda tidak dapat mengubah persyaratan TLS domain kumpulan pengguna Anda.

## Kebijakan UI dan CORS yang dihosting

UI yang dihosting Amazon Cognito tidak mendukung kebijakan asal berbagi sumber daya lintas asal (CORS) kustom. Kebijakan CORS di UI yang dihosting akan mencegah pengguna meneruskan parameter otentikasi dalam permintaan mereka. Sebagai gantinya, terapkan kebijakan CORS di frontend web aplikasi Anda. Amazon Cognito mengembalikan header `Access-Control-Allow-Origin: *` respons ke permintaan ke titik akhir OAuth berikut.

1. [Titik akhir token](#)
2. [Cabut titik akhir](#)
3. [Titik akhir UserInfo](#)

## UI yang di-host dan cookie server otorisasi

Titik akhir kumpulan pengguna Amazon Cognito mengatur cookie di browser pengguna. Cookie sesuai dengan persyaratan beberapa browser bahwa situs tidak menetapkan cookie pihak ketiga. Mereka hanya dicakup ke titik akhir kumpulan pengguna Anda dan termasuk yang berikut ini:

- `XSRF-TOKEN`Cookie untuk setiap permintaan.
- `csrf-state`Cookie untuk konsistensi sesi saat pengguna dialihkan.
- Cookie `cognito` sesi yang mempertahankan upaya masuk yang berhasil selama satu jam.

## Mengkonfigurasi domain kolam pengguna

Setelah menyiapkan klien aplikasi, Anda dapat mengkonfigurasi alamat halaman web pendaftaran dan masuk Anda. Anda dapat menggunakan domain yang dihosting Amazon Cognito dan memilih prefiks domain yang tersedia, atau Anda dapat menggunakan alamat web Anda sendiri sebagai domain kustom.

Untuk menambahkan klien aplikasi dan domain yang dihosting Amazon Cognito dengan AWS Management Console, lihat [Menambahkan aplikasi untuk mengaktifkan UI web yang dihosting](#).

**Note**

Anda tidak dapat menggunakan teksawsamazon,, ataucognito di awalan domain.

## Topik

- [Menggunakan domain Amazon Cognito untuk UI yang dihosting](#)
- [Menggunakan domain Anda sendiri untuk UI yang dihosting](#)

## Menggunakan domain Amazon Cognito untuk UI yang dihosting

Setelah menyiapkan klien aplikasi, Anda dapat mengonfigurasi alamat untuk halaman web pendaftaran dan masuk. Anda dapat menggunakan domain Amazon Cognito yang di-hosting dengan prefiks domain Anda sendiri.

**Note**

Untuk meningkatkan keamanan aplikasi Amazon Cognito Anda, domain induk dari titik akhir kumpulan pengguna terdaftar di Daftar [Akhiran Publik](#) (PSL). PSL membantu browser web pengguna Anda membangun pemahaman yang konsisten tentang titik akhir kumpulan pengguna Anda dan cookie yang mereka tetapkan.

Domain induk endpoint kumpulan pengguna mengambil format berikut.

```
auth.Region.amazoncognito.com  
auth-fips.Region.amazoncognito.com
```

Untuk menambahkan klien aplikasi dan domain yang dihosting Amazon Cognito dengan AWS Management Console, lihat. [Membuat klien aplikasi](#)

## Topik

- [Prasyarat](#)
- [Langkah 1: Konfigurasi domain pool pengguna yang dihosting](#)
- [Langkah 2: Verifikasi halaman masuk Anda](#)

## Prasyarat

Sebelum memulai, Anda memerlukan:

- Kolam pengguna dengan klien aplikasi. Untuk informasi selengkapnya, lihat [Memulai dengan kumpulan pengguna](#).

Langkah 1: Konfigurasi domain pool pengguna yang dihosting

Untuk mengonfigurasi domain kumpulan pengguna yang dihosting

Anda dapat menggunakan salah satu AWS Management Console atau AWS CLI atau API untuk mengonfigurasi domain kumpulan pengguna.

Amazon Cognito console

Mengkonfigurasi domain

1. Arahkan ke tab Integrasi aplikasi untuk kumpulan pengguna Anda.
2. Di samping Domain, pilih Tindakan dan pilih Buat domain khusus atau Buat domain Amazon Cognito. Jika Anda telah mengonfigurasi domain kumpulan pengguna, pilih Hapus domain Amazon Cognito atau Hapus domain kustom sebelum membuat domain kustom baru Anda.
3. Masukkan awalan domain yang tersedia untuk digunakan dengan domain Amazon Cognito. Untuk informasi tentang cara menyiapkan domain Kustom, lihat [Menggunakan Domain Anda sendiri untuk UI yang dihosting](#)
4. Pilih Buat.

CLI/API

Gunakan perintah berikut untuk membuat prefiks domain dan menetapkannya ke kolam pengguna Anda.

Cara mengkonfigurasi domain kolam pengguna

- AWS CLI: `aws cognito-idp create-user-pool-domain`

Contoh: `aws cognito-idp create-user-pool-domain --user-pool-id <user_pool_id> --domain <domain_name>`

- AWS API: [CreateUserPoolDomain](#)

## Cara mendapatkan informasi tentang domain

- AWS CLI: `aws cognito-idp describe-user-pool-domain`

Contoh: `aws cognito-idp describe-user-pool-domain --domain <domain_name>`

- AWS API: [DescribeUserPoolDomain](#)

## Cara menghapus domain

- AWS CLI: `aws cognito-idp delete-user-pool-domain`

Contoh: `aws cognito-idp delete-user-pool-domain --domain <domain_name>`

- AWS API: [DeleteUserPoolDomain](#)

## Langkah 2: Verifikasi halaman masuk Anda

- Verifikasi bahwa halaman masuk tersedia dari domain yang di-hosting Amazon Cognito Anda.

```
https://<your_domain>/login?  
response_type=code&client_id=<your_app_client_id>&redirect_uri=<your_callback_url>
```

Domain Anda ditampilkan di halaman Nama Domain pada konsol Amazon Cognito. ID klien aplikasi dan URL callback Anda ditampilkan di halaman Pengaturan klien aplikasi.

## Menggunakan domain Anda sendiri untuk UI yang dihosting

Setelah menyiapkan klien aplikasi, Anda dapat mengonfigurasi kumpulan pengguna dengan domain khusus untuk UI yang dihosting Amazon Cognito dan titik akhir API [auth](#). Dengan domain kustom, Anda memungkinkan pengguna untuk masuk ke aplikasi Anda dengan menggunakan alamat web Anda sendiri.

### Topik

- [Menambahkan domain khusus ke kumpulan pengguna](#)
- [Mengubah sertifikat SSL untuk domain kustom Anda](#)

## Menambahkan domain khusus ke kumpulan pengguna

Untuk menambahkan domain kustom ke kolam pengguna, tentukan nama domain di konsol Amazon Cognito, dan berikan sertifikat yang dikelola dengan [AWS Certificate Manager](#) (ACM). Setelah menambahkan domain, Amazon Cognito menyediakan target alias, yang ditambahkan ke konfigurasi DNS.

### Prasyarat

Sebelum memulai, Anda memerlukan:

- Kolam pengguna dengan klien aplikasi. Untuk informasi selengkapnya, lihat [Memulai dengan kumpulan pengguna](#).
- Domain web yang Anda miliki. Domain induknya harus memiliki catatan DNS A yang valid. Anda dapat menetapkan nilai apa pun ke catatan ini. Induk mungkin akar domain, atau domain anak yang merupakan satu langkah ke atas dalam hierarki domain. Misalnya, jika domain kustom Anda adalah `auth.xyz.example.com`, Amazon Cognito harus dapat menyelesaikan `xyz.example.com` ke alamat IP. Untuk mencegah dampak yang tidak disengaja pada infrastruktur pelanggan, Amazon Cognito tidak mendukung penggunaan domain tingkat atas (TLD) untuk domain kustom. Untuk informasi selengkapnya lihat [Nama Domain](#).
- Kemampuan untuk membuat subdomain untuk domain kustom Anda. Kami merekomendasikan penggunaan `auth` sebagai subdomain. Misalnya: `auth.example.com`.

#### Note

Jika Anda tidak memiliki [sertifikat wildcard](#), Anda harus mendapatkan sertifikat baru untuk subdomain domain kustom Anda.

- Sertifikat Secure Sockets Layer (SSL) yang dikelola oleh ACM.

#### Note

Anda harus mengubah AWS wilayah ke US East (Virginia N.) di konsol ACM sebelum Anda meminta atau mengimpor sertifikat.

- Aplikasi yang memungkinkan server otorisasi kumpulan pengguna Anda untuk menambahkan cookie ke sesi pengguna. Amazon Cognito menetapkan beberapa cookie yang diperlukan untuk UI yang dihosting. Ini termasuk `cognito`, `cognito-f1`, dan `XSRF-TOKEN`. Meskipun setiap cookie

individu sesuai dengan batas ukuran browser, perubahan pada konfigurasi kumpulan pengguna Anda dapat menyebabkan cookie UI yang dihosting bertambah besar. Layanan perantara seperti Application Load Balancer (ALB) di depan domain kustom Anda dapat menerapkan ukuran header maksimum atau ukuran cookie total. Jika aplikasi Anda juga menetapkan cookie sendiri, sesi pengguna Anda mungkin melebihi batas ini. Kami menyarankan agar, untuk menghindari konflik batas ukuran, aplikasi Anda tidak menetapkan cookie pada subdomain UI yang dihosting.

- Izin untuk memperbarui CloudFront distribusi Amazon. Anda dapat melakukannya dengan melampirkan pernyataan kebijakan IAM berikut kepada pengguna di: Akun AWS

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowCloudFrontUpdateDistribution",
      "Effect": "Allow",
      "Action": [
        "cloudfront:updateDistribution"
      ],
      "Resource": [
        "*"
      ]
    }
  ]
}
```

Untuk informasi selengkapnya tentang otorisasi tindakan CloudFront, lihat [Menggunakan Kebijakan Berbasis Identitas \(Kebijakan IAM\)](#) untuk CloudFront

Amazon Cognito awalnya menggunakan izin IAM Anda untuk mengonfigurasi CloudFront distribusi, tetapi distribusi dikelola oleh AWS Anda tidak dapat mengubah konfigurasi CloudFront distribusi yang dikaitkan Amazon Cognito dengan kumpulan pengguna Anda. Misalnya, Anda tidak dapat memperbarui versi TLS yang didukung dalam kebijakan keamanan.

### Langkah 1: Masukkan nama domain kustom Anda

Anda dapat menambahkan domain ke kolam pengguna dengan menggunakan konsol Amazon Cognito atau API.

## Amazon Cognito console

Untuk menambahkan domain ke kumpulan pengguna dari konsol Amazon Cognito:

1. Masuk ke [konsol Amazon Cognito](#). Jika diminta, masukkan AWS kredensial Anda.
2. Pilih kumpulan Pengguna.
3. Pilih kumpulan pengguna yang ingin Anda tambahkan domain Anda.
4. Pilih tab Integrasi aplikasi.
5. Di sebelah Domain, pilih Tindakan, lalu pilih Buat domain khusus.

### Note

Jika Anda telah mengonfigurasi domain kumpulan pengguna, pilih Hapus domain Cognito atau Hapus domain khusus untuk menghapus domain yang ada sebelum membuat domain kustom baru Anda.

6. Untuk Domain kustom, masukkan URL domain yang ingin Anda gunakan dengan Amazon Cognito. Nama domain Anda hanya dapat menyertakan huruf kecil, angka, dan tanda hubung. Jangan gunakan tanda hubung untuk karakter pertama atau terakhir. Gunakan titik untuk memisahkan nama subdomain.
7. Untuk sertifikat ACM, pilih sertifikat SSL yang ingin Anda gunakan untuk domain Anda. Hanya sertifikat ACM di US East (Virginia N.) yang memenuhi syarat untuk digunakan dengan domain khusus Amazon Cognito, terlepas dari kumpulan pengguna Wilayah AWS Anda.

Jika Anda tidak memiliki sertifikat yang tersedia, Anda dapat menggunakan ACM untuk menyediakannya di AS Timur (Virginia N.). Untuk informasi lebih lanjut, lihat [Memulai](#) di AWS Certificate Manager Panduan Pengguna.

8. Pilih Buat.
9. Amazon Cognito mengembalikan Anda ke tab Integrasi aplikasi. Pesan berjudul Buat catatan alias di DNS domain Anda ditampilkan. Catat target Domain dan Alias yang ditampilkan di konsol. Mereka akan digunakan pada langkah berikutnya untuk mengarahkan lalu lintas ke domain kustom Anda.

## API

Untuk menambahkan domain Anda ke kumpulan pengguna Anda dengan Amazon Cognito API:

- Gunakan [CreateUserPoolDomain](#) tindakan.

### Langkah 2: Tambahkan target alias dan subdomain

Pada langkah ini, Anda mengatur alias melalui penyedia layanan Domain Name Server (DNS) yang menunjuk kembali ke target alias dari langkah sebelumnya. Jika Anda menggunakan Amazon Route 53 untuk resolusi alamat DNS, pilih bagian Untuk menambahkan target alias dan subdomain dengan menggunakan Route 53.

Cara menambahkan target alias dan subdomain ke konfigurasi DNS Anda saat ini

- Jika Anda tidak menggunakan Route 53 untuk resolusi alamat DNS, maka Anda harus menggunakan alat konfigurasi penyedia layanan DNS Anda untuk menambahkan target alias dari langkah sebelumnya ke catatan DNS domain Anda. Penyedia DNS Anda juga perlu mengatur subdomain untuk domain kustom Anda.

Cara menambahkan target alias dan subdomain yang menggunakan Route 53

1. Masuk ke [Konsol Route 53](#). Jika diminta, masukkan AWS kredensial Anda.
2. Jika Anda tidak memiliki zona yang dihosting di Route 53, buat zona dengan root yang merupakan induk dari domain khusus Anda. Untuk informasi selengkapnya, silakan lihat
  - a. Pilih Buat Zona yang Di-hosting.
  - b. Masukkan domain induk, misalnya ***auth.example.com***, domain kustom Anda, misalnya ***myapp.auth.example.com***, dari daftar Nama Domain.
  - c. Masukkan Deskripsi untuk zona host Anda.
  - d. Pilih zona yang dihosting Jenis zona yang dihosting publik untuk memungkinkan klien publik menyelesaikan domain kustom Anda. Memilih Zona yang dihosting pribadi tidak didukung.
  - e. Terapkan Tag sesuai keinginan.
  - f. Pilih Buat zona yang di-hosting.



**Note**

Anda juga dapat membuat zona host baru untuk domain kustom Anda, dan Anda dapat membuat set delegasi di zona host induk yang mengarahkan kueri ke zona host subdomain. Jika tidak, buat catatan A. Metode ini menawarkan lebih banyak fleksibilitas dan keamanan dengan zona yang dihosting Anda. Untuk informasi selengkapnya, lihat [Membuat subdomain untuk domain yang dihosting melalui Amazon Route 53](#).

3. Pada halaman Zona yang Di-Hosting, pilih nama zona yang di-hosting.
4. Tambahkan catatan DNS untuk domain induk domain kustom Anda, jika Anda belum memilikinya. Tambahkan A catatan DNS untuk domain induk dan pilih Buat catatan. Berikut ini adalah contoh catatan untuk domain *auth.example.com*.

```
auth.example.com. 60 IN A 198.51.100.1
```

**Note**

Amazon Cognito memverifikasi bahwa ada catatan DNS untuk domain induk domain kustom Anda untuk melindungi dari pembajakan domain produksi yang tidak disengaja. Jika Anda tidak memiliki catatan DNS untuk domain induk, Amazon Cognito akan menampilkan kesalahan saat Anda mencoba menyetel domain kustom. Catatan Start of Authority (SOA) bukanlah catatan DNS yang cukup untuk tujuan verifikasi domain induk.

5. Tambahkan catatan DNS untuk domain kustom Anda. Anda merekam harus menunjuk ke target Alias domain kustom, misalnya *123example.cloudfront.net*. Pilih Buat catatan lagi.
6. Masukkan nama Rekam yang cocok dengan domain kustom Anda, misalnya *myapp untuk membuat catatan untuk myapp.auth.example.com*.
7. Aktifkan opsi Alias.
8. Pilih untuk Merutekan lalu lintas ke distribusi Alias ke Cloudfront. Masukkan target Alias yang disediakan oleh Amazon Cognito saat Anda membuat domain khusus.
9. Pilih Buat Catatan.

**Note**

Catatan baru Anda dapat memakan waktu sekitar 60 detik untuk disebarakan ke semua server DNS Route 53. Anda dapat menggunakan metode Route 53 [GetChangeAPI](#) untuk memverifikasi bahwa perubahan Anda telah disebarakan.

**Langkah 3: Verifikasi halaman masuk Anda**

- Verifikasi bahwa halaman masuk tersedia dari domain kustom Anda.

Masuk dengan domain dan subdomain kustom Anda dengan memasukkan alamat ini ke browser Anda. Ini adalah contoh URL dari domain kustom *example.com* dengan subdomain *auth*:

```
https://myapp.auth.example.com/login?  
response_type=code&client_id=<your_app_client_id>&redirect_uri=<your_callback_url>
```

**Mengubah sertifikat SSL untuk domain kustom Anda**

Bila diperlukan, Anda dapat menggunakan Amazon Cognito untuk mengubah sertifikat yang Anda terapkan ke domain kustom Anda.

Biasanya, ini tidak diperlukan setelah pembaruan sertifikat rutin dilakukan dengan ACM. Ketika Anda memperbarui sertifikat yang ada di ACM, ARN untuk sertifikat Anda tetap sama, dan domain kustom Anda menggunakan sertifikat baru secara otomatis.

Namun, jika Anda mengganti sertifikat yang sudah ada dengan yang baru, ACM memberikan ARN baru ke sertifikat baru. Untuk menerapkan sertifikat baru ke domain kustom Anda, Anda harus memberikan ARN ini ke Amazon Cognito.

Setelah Anda memberikan sertifikat baru, Amazon Cognito memerlukan waktu hingga 1 jam untuk mendistribusikannya ke domain kustom Anda.

**Sebelum Anda memulai**

Sebelum Anda dapat mengubah sertifikat Anda di Amazon Cognito, Anda harus menambahkan sertifikat Anda ke ACM. Untuk informasi lebih lanjut, lihat [Memulai](#) di AWS Certificate Manager Panduan Pengguna.

Ketika Anda menambahkan sertifikat ke ACM, Anda harus memilih US East (N. Virginia) sebagai Wilayah AWS .

Anda dapat mengubah sertifikat dengan menggunakan konsol Amazon Cognito atau API.

## AWS Management Console

Untuk memperbarui sertifikat dari konsol Amazon Cognito:

1. Masuk ke AWS Management Console dan buka konsol Amazon Cognito di <https://console.aws.amazon.com/cognito/home>
2. Pilih Kolam Pengguna.
3. Pilih kumpulan pengguna yang ingin Anda perbarui sertifikatnya.
4. Pilih tab Integrasi aplikasi.
5. Pilih Tindakan, Edit sertifikat ACM.
6. Pilih sertifikat baru yang ingin Anda kaitkan dengan domain kustom Anda.
7. Pilih Simpan perubahan.

## API

Cara memperbarui sertifikat (API Amazon Cognito)

- Gunakan [UpdateUserPoolDomain](#)tindakan.

## Menyesuaikan halaman web masuk dan daftar bawaan

Anda dapat menggunakan AWS Management Console, atau AWS CLI atau API, untuk menentukan pengaturan penyesuaian untuk pengalaman UI aplikasi bawaan. Anda dapat mengunggah gambar logo kustom yang akan ditampilkan di aplikasi. Anda juga dapat menggunakan cascading style sheet (CSS) untuk menyesuaikan tampilan UI.

Anda dapat menentukan pengaturan kustomisasi UI aplikasi untuk satu klien (dengan `clientId` spesifik) atau untuk semua klien (dengan menetapkan `clientId` ke ALL). Jika Anda menentukan ALL, konfigurasi default akan digunakan untuk setiap klien yang tidak memiliki set kustomisasi UI sebelumnya. Jika Anda menentukan pengaturan kustomisasi UI untuk klien tertentu, itu tidak akan kembali lagi ke konfigurasi ALL.

Permintaan yang menetapkan kustomisasi UI Anda tidak boleh melebihi 135 KB. Dalam kasus yang jarang terjadi, jumlah header permintaan, file CSS Anda, dan logo Anda mungkin melebihi 135 KB. Amazon Cognito mengkodekan file gambar ke Base64. Ini meningkatkan ukuran gambar 100 KB menjadi 130 KB, menyimpan lima KB untuk header permintaan dan CSS Anda. Jika permintaan terlalu besar, permintaan `SetUICustomization` API AWS Management Console atau Anda mengembalikan `request parameters too large` kesalahan. Sesuaikan gambar logo Anda menjadi tidak lebih dari 100KB dan file CSS Anda tidak lebih besar dari 3 KB. Anda tidak dapat mengatur CSS dan kustomisasi logo secara terpisah.

#### Note

Untuk menyesuaikan UI Anda, Anda harus mengatur domain untuk kolom pengguna Anda.

## Menentukan logo kustom untuk aplikasi

Amazon Cognito memusatkan logo kustom Anda di atas kolom input di [Titik akhir masuk](#)

Pilih file PNG, JPG, atau JPEG yang dapat menskalakan hingga 350 x 178 piksel untuk logo UI yang di-host khusus Anda. File logo Anda dapat berukuran tidak lebih dari 100 KB, atau 130 KB setelah Amazon Cognito dikodekan ke Base64. Untuk menyetel `ImageFile` [SetUICustomization](#) dalam API, konversikan file Anda ke string teks yang dikodekan Base64 atau, diAWS CLI, sediakan jalur file dan biarkan Amazon Cognito menyandikannya untuk Anda.

## Menentukan kustomisasi CSS untuk aplikasi

Anda dapat mengkustomisasi CSS untuk halaman aplikasi yang di-hosting, dengan pembatasan berikut:

- Anda dapat menggunakan salah satu nama kelas CSS berikut:
  - `background-customizable`
  - `banner-customizable`
  - `errorMessage-customizable`
  - `idpButton-customizable`
  - `idpButton-customizable: hover`
  - `idpDescription-customizable`
  - `inputField-customizable`

- `inputField-customizable:focus`
- `label-customizable`
- `legalText-customizable`
- `logo-customizable`
- `passwordCheck-valid-customizable`
- `passwordCheck-notValid-customizable`
- `redirect-customizable`
- `socialButton-customizable`
- `submitButton-customizable`
- `submitButton-customizable:hover`
- `textDescription-customizable`
- Nilai properti dapat berisi HTML, kecuali untuk nilai-nilai berikut: `@import`, `@supports@page`, atau `@media` pernyataan, atau Javascript.

Anda dapat mengkustomisasi properti CSS berikut.

#### Label

- bobot huruf adalah kelipatan 100 dari 100 sampai 900.

#### Bidang input

- lebar blok yang menampungnya dalam persentase.
- tinggi adalah tinggi bidang input dalam piksel (px).
- warna adalah warna teks. Ini bisa berupa setiap nilai warna CSS standar.
- warna latar belakang adalah warna latar belakang bidang input. Ini bisa berupa setiap nilai warna CSS standar.
- perbatasan adalah nilai batas CSS standar yang menentukan lebar, transparansi, dan warna perbatasan jendela aplikasi Anda. Lebar dapat berupa nilai dari 1px hingga 100px. Transparansi bisa jadi padat atau tidak ada. Warna bisa berupa setiap nilai warna standar.

#### Deskripsi teks

- `padding-top` adalah jumlah padding di atas deskripsi teks.
- `padding-bottom` adalah jumlah padding di bawah deskripsi teks.
- tampilan dapat berupa `block` atau `inline`.
- ukuran huruf adalah ukuran huruf untuk deskripsi teks.

## Tombol kirim

- ukuran huruf adalah ukuran huruf pada teks tombol.
- bobot huruf adalah bobot huruf pada teks tombol: `bold`, `italic`, atau `normal`.
- margin adalah string dari empat nilai yang menunjukkan ukuran margin atas, kanan, bawah, dan kiri untuk tombol.
- ukuran huruf adalah ukuran huruf untuk deskripsi teks.
- lebar adalah lebar teks tombol dalam persentase pada blok yang menampungnya.
- tinggi adalah tinggi tombol dalam piksel (px).
- warna adalah warna teks tombol. Ini bisa berupa setiap nilai warna CSS standar.
- warna latar belakang adalah warna latar belakang tombol. Ini bisa berupa setiap nilai warna standar.

## Banner

- padding adalah string dari empat nilai yang menunjukkan ukuran padding atas, kanan, bawah, dan kiri untuk banner.
- warna latar belakang adalah warna latar belakang banner. Ini bisa berupa setiap nilai warna CSS standar.

## Pengarahan tombol kirim

- warna adalah warna latar depan tombol saat Anda mengarahkan kursor ke atasnya. Ini bisa berupa setiap nilai warna CSS standar.
- warna latar belakang adalah warna latar belakang tombol saat Anda mengarahkan kursor ke atasnya. Ini bisa berupa setiap nilai warna CSS standar.

## Pengarahan tombol penyedia identitas

- warna adalah warna latar depan tombol saat Anda mengarahkan kursor ke atasnya. Ini bisa berupa setiap nilai warna CSS standar.
- warna latar belakang adalah warna latar belakang tombol saat Anda mengarahkan kursor ke atasnya. Ini bisa berupa setiap nilai warna CSS standar.

## Cek kata sandi tidak valid

- warna adalah warna teks pesan "`Password check not valid`". Ini bisa berupa setiap nilai warna CSS standar.

## Latar Belakang

- warna latar belakang adalah warna latar belakang pada jendela aplikasi. Ini bisa berupa setiap nilai warna CSS standar.

## Pesan kesalahan

- margin adalah string dari empat nilai yang menunjukkan ukuran margin atas, kanan, bawah, dan kiri.
- padding adalah ukuran padding.
- Ukuran huruf adalah ukuran huruf.
- lebar adalah lebar pesan kesalahan dalam persentase pada blok yang menampungnya.
- latar belakang adalah warna latar belakang pada pesan kesalahan. Ini bisa berupa setiap nilai warna CSS standar.
- perbatasan adalah string dari tiga nilai menentukan lebar, transparansi, dan warna perbatasan.
- warna adalah warna teks pesan kesalahan. Ini bisa berupa setiap nilai warna CSS standar.
- ukuran kotak digunakan untuk menunjukkan kepada browser apa yang harus disertakan oleh properti ukuran (lebar dan tinggi).

## Tombol penyedia identitas

- tinggi adalah tinggi tombol dalam piksel (px).
- lebar adalah lebar teks tombol dalam bentuk persentase dari blok yang menampungnya.
- rata-teks adalah pengaturan perataan teks. Status tersebut dapat berupa `left`, `right`, atau `center`.
- margin-bawah adalah pengaturan margin bawah.
- warna adalah warna teks tombol. Ini bisa berupa setiap nilai warna CSS standar.
- warna latar belakang adalah warna latar belakang tombol. Ini bisa berupa setiap nilai warna CSS standar.
- warna batas adalah warna batas tombol. Ini bisa berupa setiap nilai warna CSS standar.

## Deskripsi penyedia identitas

- padding-top adalah jumlah padding di atas deskripsi.
- padding-bottom adalah jumlah padding di bawah deskripsi.
- tampilan dapat berupa `block` atau `inline`.
- ukuran huruf adalah ukuran huruf untuk deskripsi.

## Teks legal

- warna adalah warna teks. Ini bisa berupa setiap nilai warna CSS standar.
- Ukuran huruf adalah ukuran huruf.

**Note**

Saat Anda menyesuaikan teks Legal, Anda menyesuaikan pesan Kami tidak akan memposting ke akun Anda tanpa meminta terlebih dahulu yang ditampilkan di bawah penyedia identitas sosial di halaman login.

**Logo**

- lebar maksimal adalah lebar maksimal dalam persentase pada blok yang menampungnya.
- tinggi maksimal adalah tinggi maksimal dalam persentase pada blok yang menampungnya.

**Fokus bidang input**

- warna batas adalah warna dari bidang input. Ini bisa berupa setiap nilai warna CSS standar.
- garis batas adalah lebar perbatasan bidang input, dalam piksel.

**Tombol sosial**

- tinggi adalah tinggi tombol dalam piksel (px).
- rata-teks adalah pengaturan perataan teks. Status tersebut dapat berupa `left`, `right`, atau `center`.
- lebar adalah lebar teks tombol dalam bentuk persentase dari blok yang menampungnya.
- margin-bawah adalah pengaturan margin bawah.

**Pemeriksaan kata sandi valid**

- warna adalah warna teks pesan "Password check valid". Ini bisa berupa setiap nilai warna CSS standar.

**Menentukan pengaturan kustomisasi UI untuk kolam pengguna () AWS Management Console**

Anda dapat menggunakan AWS Management Console untuk menentukan pengaturan kustomisasi UI untuk aplikasi Anda.

**Note**

Anda dapat melihat UI yang dihosting dengan penyesuaian Anda dengan membangun URL berikut, dengan spesifik untuk kolam pengguna Anda, dan mengetikkannya ke peramban: `https://<your_domain>/login?`





- AWSAPI: [GetUICustomization](#)

Cara menyetel pengaturan kustomisasi kustomisasi UI untuk UI kustomisasi kustomisasi UI untuk UI kustomisasi kustomisasi UI untuk UI kustomisasi kustomisasi UI untuk UI kustomisasi UI untuk

- AWS CLI dari file gambar: 

```
aws cognito-idp set-ui-customization --user-pool-id <your-user-pool-id> --client-id <your-app-client-id> --image-file fileb://<path-to-logo-image-file> --css ".label-customizable{ color: <color>;}"
```
- AWS CLI dengan gambar yang dikodekan sebagai teks biner Base64: 

```
aws cognito-idp set-ui-customization --user-pool-id <your-user-pool-id> --client-id <your-app-client-id> --image-file <base64-encoded-image-file> --css ".label-customizable{ color: <color>;}"
```
- AWS API: [SetUICustomization](#)

## Mendaftar dan masuk dengan UI yang dihosting

Setelah mengonfigurasi dan menyesuaikan UI yang dihosting Amazon Cognito untuk kumpulan pengguna dan klien aplikasi Anda, aplikasi Anda dapat menyajikannya kepada pengguna Anda. UI yang dihosting mendukung beberapa operasi autentikasi Amazon Cognito, termasuk contoh berikut.

- Mendaftar sebagai pengguna baru di aplikasi Anda
- Verifikasi alamat email atau nomor telepon
- Mengatur Autentikasi Multi-Factor (MFA)
- Masuk dengan nama pengguna dan kata sandi lokal
- Masuk dengan penyedia identitas pihak ketiga (IdP)
- Mengatur ulang kata sandi

UI yang dihosting Amazon Cognito dimulai di [Titik akhir masuk](#) URL ke halaman login adalah kombinasi dari domain yang Anda pilih untuk pangkalan pengguna, dan parameter yang mencerminkan hibah OAuth 2.0 yang ingin Anda terbitkan, klien aplikasi Anda, jalur ke aplikasi Anda, dan cakupan OpenID Connect (OIDC) yang ingin Anda minta.

```
https://<your user pool domain>/authorize?client_id=<your app client ID>&response_type=<code/token>&scope=<scopes to request>&redirect_uri=<your callback URL>
```

URL berikut menggantikan bidang placeholder di atas dengan nilai contoh.

```
https://auth.example.com/authorize? /
client_id=1example23456789 /
&response_type=code /
&scope=aws.cognito.signin.user.admin+email+openid+profile /
&redirect_uri=https%3A%2F%2Faws.amazon.com
```

Halaman login untuk UI yang dihosting Amazon Cognito memiliki opsi untuk masuk melalui pangkalan pengguna atau penyedia identitas apa pun (IdPs) yang Anda tetapkan ke klien aplikasi yang diminta pengguna Anda. Ini juga mencakup tautan untuk mendaftar akun pengguna baru di kumpulan pengguna atau untuk mengatur ulang kata sandi yang terlupakan.

Sign in with your corporate ID

MYSSO

Sign In with your social account

Continue with Apple

Continue with Login with Amazon

Continue with Google

Continue with Facebook

We won't post to any of your accounts without asking first

Sign in with your username and password

Username

Password

OR

Forgot your password?

Sign in

Need an account? [Sign up](#)

## Topik

- [Cara mendaftarkan akun baru di UI yang dihosting Amazon Cognito](#)
- [Cara masuk dengan UI yang dihosting Amazon Cognito](#)
- [Cara mengatur ulang kata sandi dengan UI yang dihosting Amazon Cognito](#)

## Cara mendaftarkan akun baru di UI yang dihosting Amazon Cognito

Panduan ini menunjukkan cara mendaftarkan akun pengguna di aplikasi yang menggunakan Amazon Cognito.

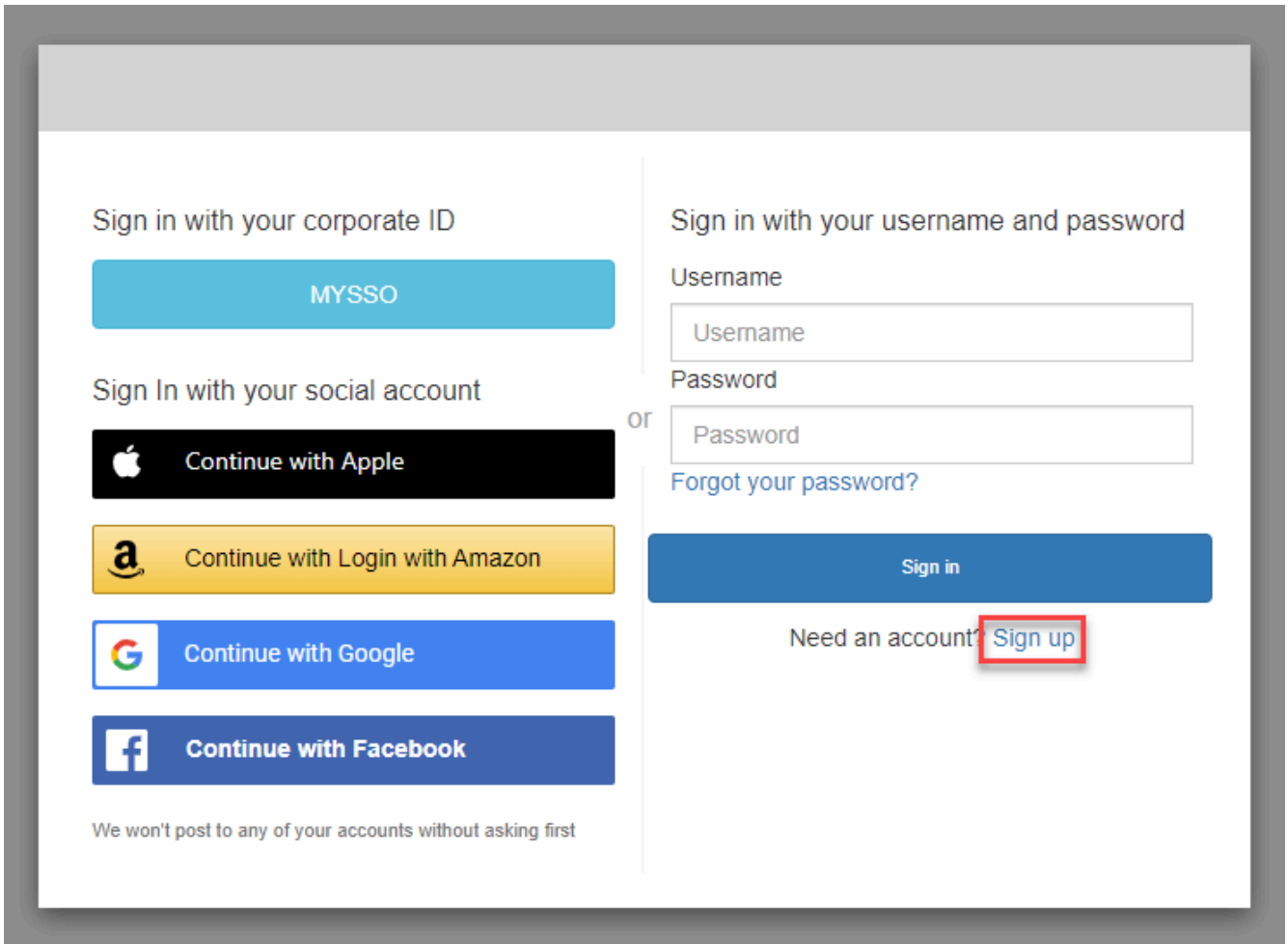
 Note

Saat masuk ke aplikasi yang menggunakan antarmuka pengguna (UI) yang dihosting Amazon Cognito, Anda mungkin melihat halaman yang telah disesuaikan oleh pemilik aplikasi di luar konfigurasi dasar yang ditunjukkan dalam panduan ini.

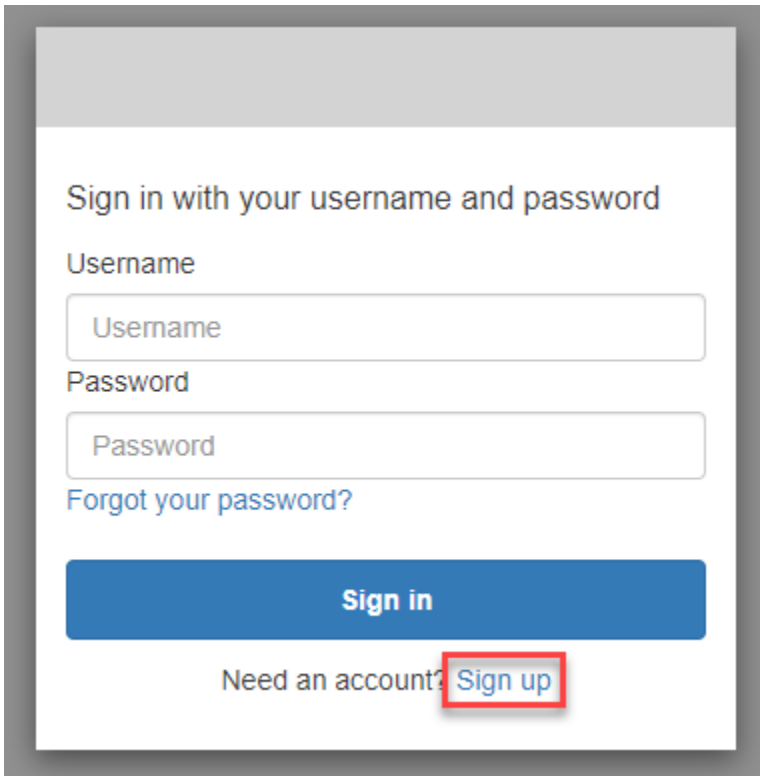
1. Pilih Daftar dari halaman login jika Anda ingin masuk melalui Amazon Cognito dengan nama pengguna dan kata sandi, bukan salah satu penyedia login pihak ketiga yang telah terdaftar oleh pemilik aplikasi.

Jika penyedia login Anda selain Amazon Cognito, pendaftaran Anda selesai setelah Anda memilih tombol untuk penyedia pihak ketiga Anda. Bergantung pada opsi yang dipilih pemilik aplikasi, Anda mungkin memiliki pilihan penyedia untuk masuk, atau Anda mungkin hanya melihat prompt untuk nama pengguna dan kata sandi.

## With multiple sign-in providers



## With only Amazon Cognito as a sign-in provider



Sign in with your username and password

Username

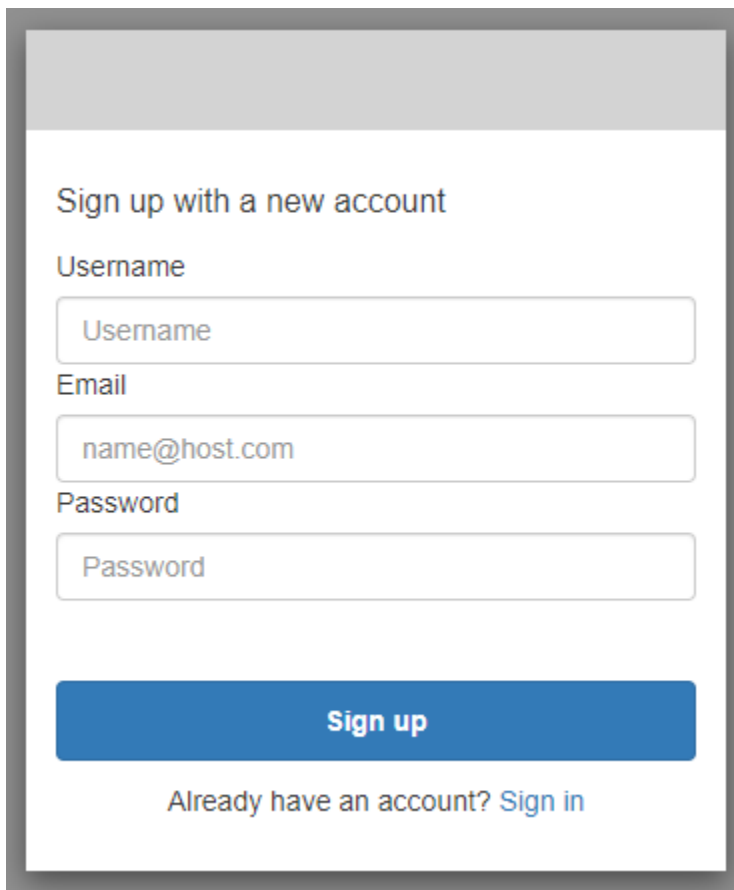
Password

[Forgot your password?](#)

**Sign in**

Need an account? [Sign up](#)

2. Pada halaman Daftar dengan akun baru, pemilik aplikasi meminta informasi yang mereka perlukan untuk mendaftar. Mereka mungkin meminta nama pengguna, alamat email, atau nomor telepon. Masukkan informasi yang diperlukan dan pilih kata sandi.



Sign up with a new account

Username

Email

Password

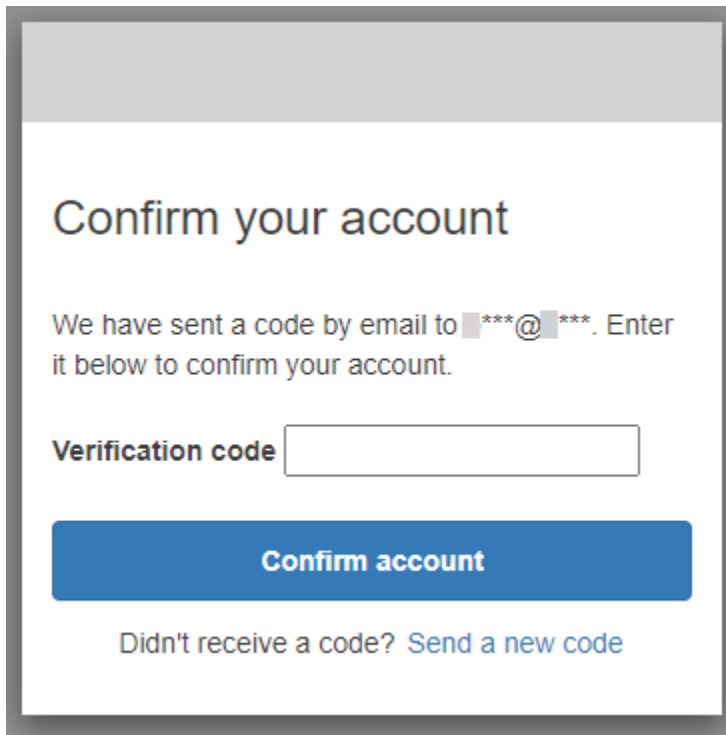
**Sign up**

Already have an account? [Sign in](#)

3. Pada halaman Konfirmasi akun Anda, pemilik aplikasi mungkin meminta Anda mengonfirmasi akun untuk memverifikasi bahwa Anda dapat menerima pesan di alamat email atau nomor telepon yang Anda berikan.

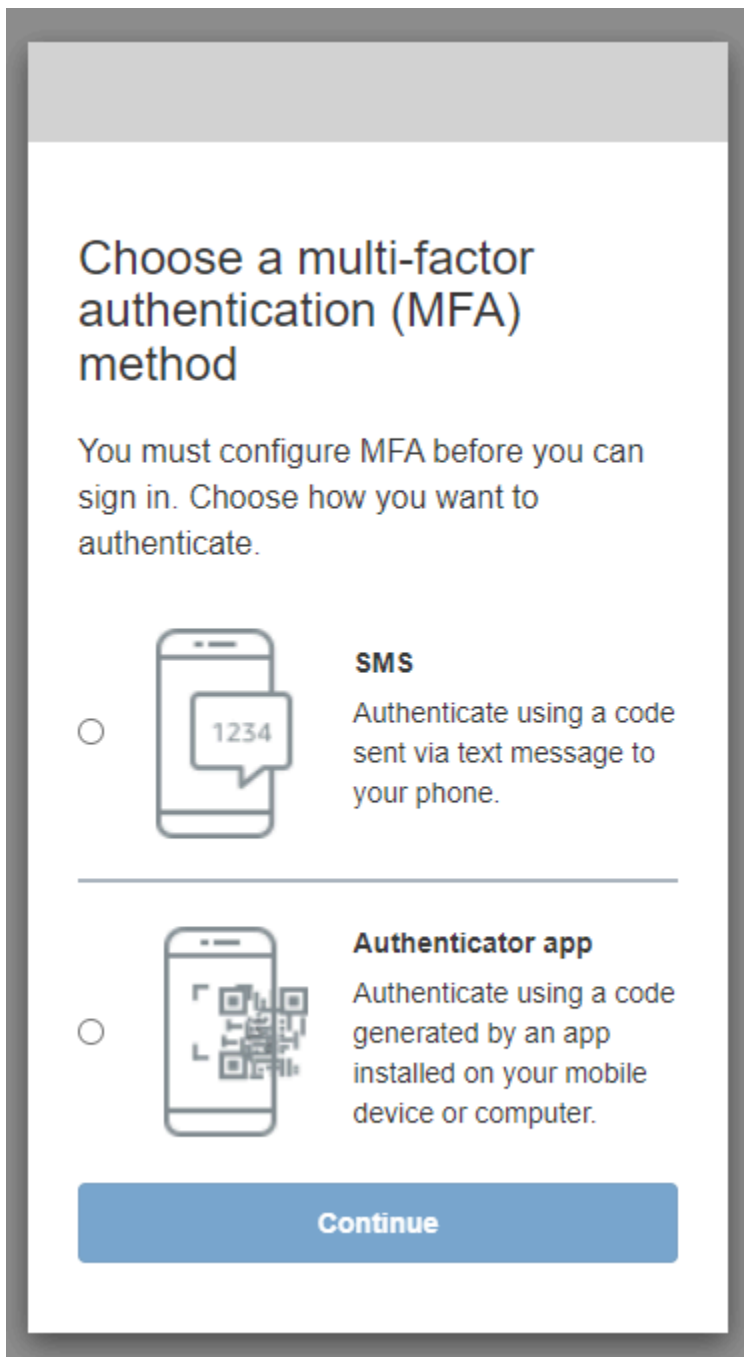
Anda akan menerima kode di email atau pesan teks SMS. Masukkan kode dalam formulir untuk mengonfirmasi bahwa Anda memasukkan informasi kontak yang benar.





4. Pemilik aplikasi mungkin mengharuskan Anda mengatur Multi-Factor Authentication (MFA). Anda mungkin melihat prompt untuk memilih metode MFA Anda, atau aplikasi Anda mungkin melompat ke langkah berikutnya.

Pada halaman Pilih metode Multi-Factor Authentication (MFA), pilih metode MFA. Jika Anda memilih SMS, Anda akan menerima kode sandi MFA dalam pesan teks SMS. Jika memilih aplikasi Authenticator, Anda harus menginstal aplikasi di perangkat untuk menghasilkan kode sandi MFA berbasis waktu. Anda harus membuat pilihan dalam waktu 3 menit.



5. Amazon Cognito meminta kode dari aplikasi autentikator atau pesan teks SMS. Masukkan kode yang Anda terima dalam waktu 3 menit.


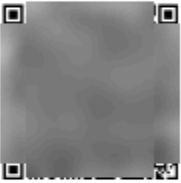
#### Authenticator app

1. Buka aplikasi autentikator yang Anda unduh.
2. Pindai kode QR pada halaman dengan kamera Anda. Anda mungkin perlu mengotorisasi aplikasi untuk menggunakan kamera Anda.

Jika Anda tidak dapat memindai kode QR, pilih Tampilkan kunci rahasia untuk menampilkan kode yang dapat Anda masukkan secara manual ke aplikasi autentikator Anda.

3. Aplikasi autentikator Anda mulai menampilkan kode yang berubah setiap beberapa detik. Masukkan kode saat ini dari aplikasi.
4. (Opsional) Pada halaman Mengatur Aplikasi Authenticator, pilih nama untuk perangkat Anda. Saat Anda masuk, Amazon Cognito akan meminta kode dari perangkat dengan nama yang Anda berikan di sini.

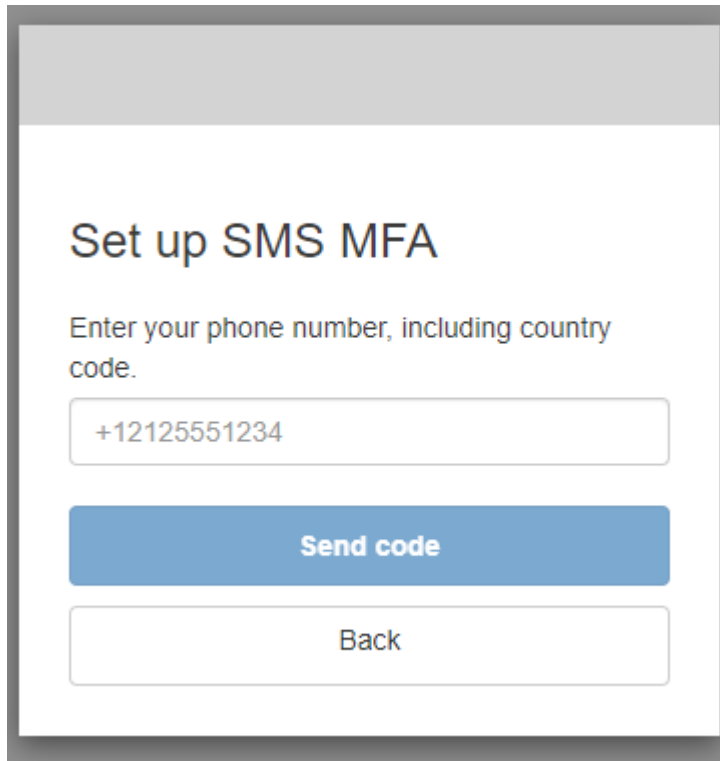
## Set up authenticator app MFA

-  Install an authenticator app on your mobile device.
-  Scan this QR code with your authenticator app. Alternatively, you can manually enter a secret key in your authenticator app.  
[Show secret key](#)
- Enter a code from your authenticator app  
  
Enter a friendly device name - optional

## SMS text message

1. Jika pemilik aplikasi belum mengumpulkan nomor telepon Anda, Amazon Cognito meminta nomor telepon Anda.

Pada halaman Atur SMS MFA, masukkan nomor telepon yang menyertakan + tanda dan kode negara, misalnya+12125551234.



Set up SMS MFA

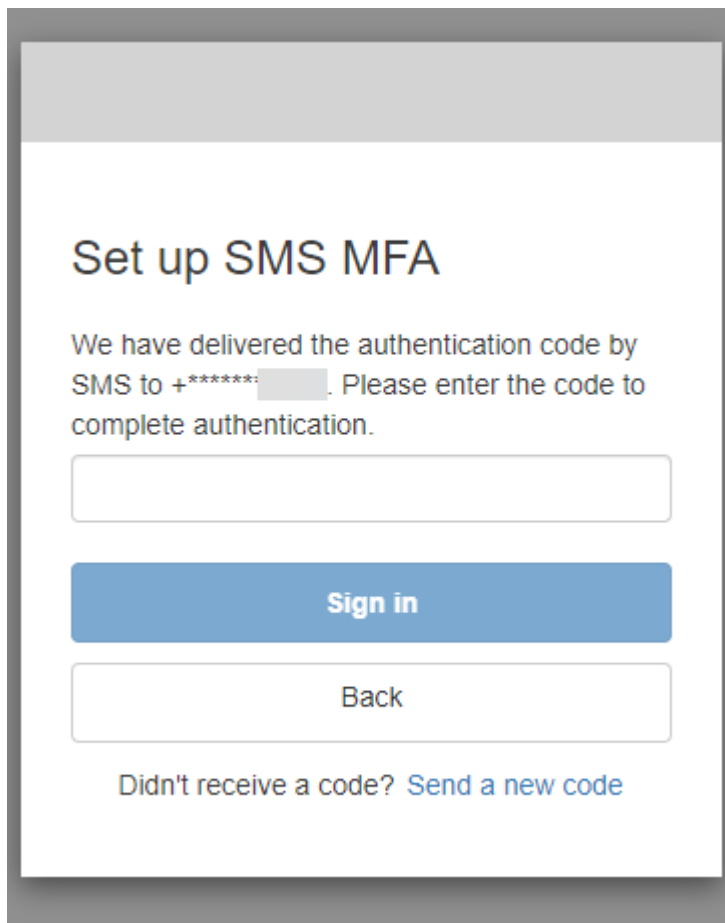
Enter your phone number, including country code.

+12125551234

Send code

Back

2. Anda akan menerima pesan SMS dengan kode. Pada halaman Set up SMS MFA, masukkan kode. Jika Anda tidak mendapatkan kode dan Anda ingin mencoba lagi, pilih Kirim kode baru. Pilih Kembali untuk memasukkan nomor telepon baru.



6. Saat pertama kali Anda mendaftar dan mengonfirmasi detail Anda, Amazon Cognito memberikan akses ke aplikasi Anda setelah Anda menyelesaikan proses ini.

## Cara masuk dengan UI yang dihosting Amazon Cognito

Panduan ini menunjukkan cara masuk ke aplikasi yang menggunakan Amazon Cognito.

### Note

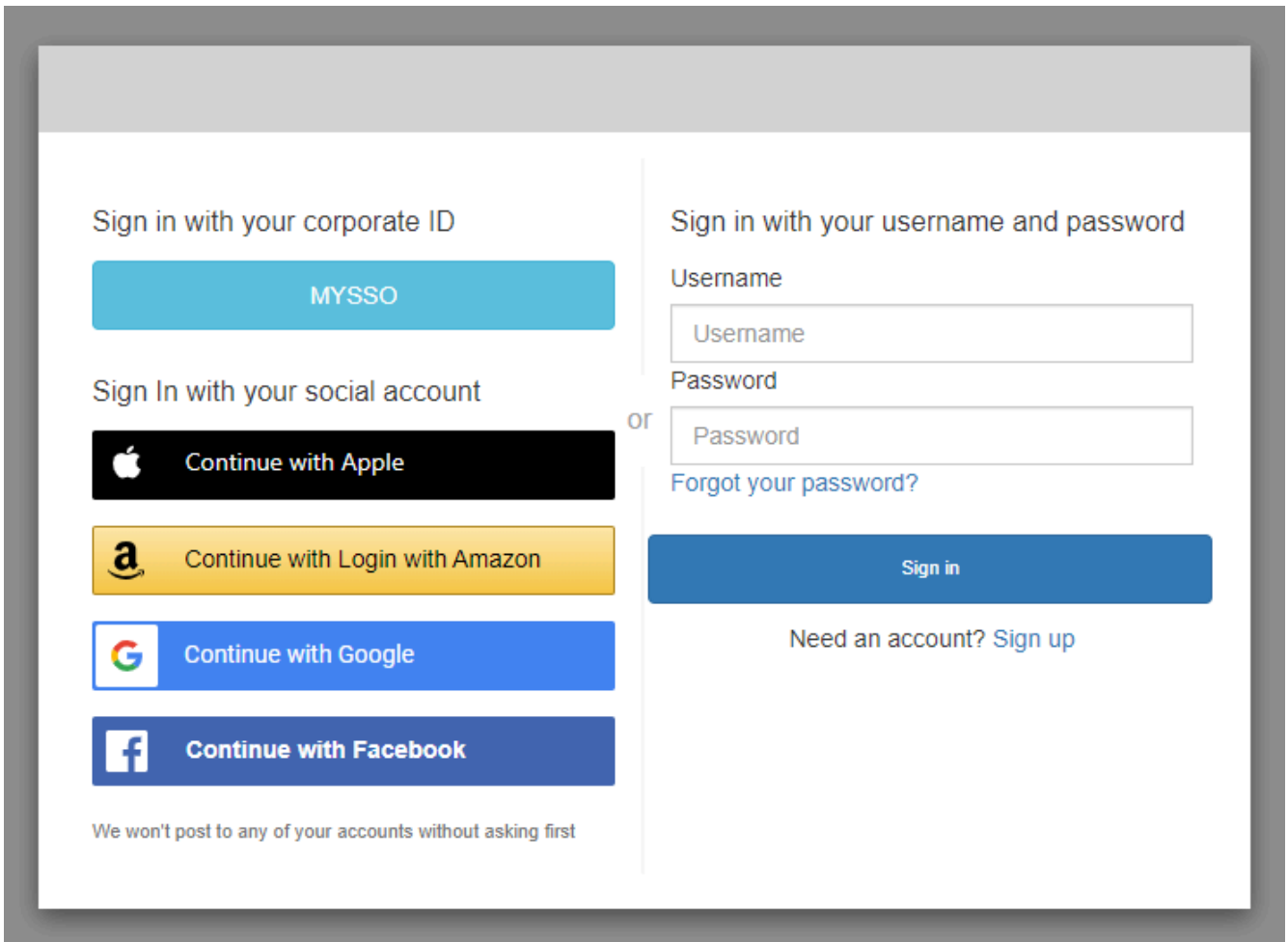
Saat masuk ke aplikasi yang menggunakan antarmuka pengguna (UI) yang dihosting Amazon Cognito, Anda mungkin melihat halaman yang telah disesuaikan oleh pemilik aplikasi di luar konfigurasi dasar yang ditunjukkan dalam panduan ini.

1. Bergantung pada opsi yang dipilih pemilik aplikasi, Anda mungkin memiliki pilihan penyedia untuk masuk, atau Anda mungkin hanya melihat prompt untuk nama pengguna dan kata sandi.

Saat Anda masuk dengan nama pengguna dan sandi dari halaman ini, Amazon Cognito adalah penyedia login Anda. Jika tidak, penyedia login Anda diwakili oleh tombol yang Anda pilih.

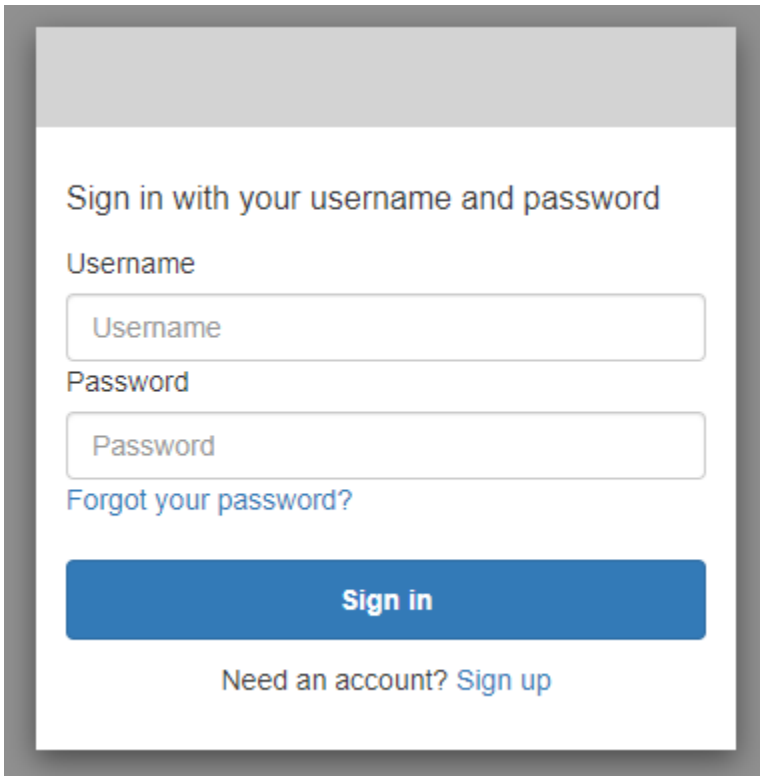
Anda dapat memilih penyedia di sini, atau memasukkan nama pengguna dan kata sandi, dan segera mendapatkan akses ke aplikasi Anda. Jika Amazon Cognito adalah penyedia login Anda, pemilik aplikasi mungkin juga memerlukan autentikasi multi-faktor.

With multiple sign-in providers



The image shows a sign-in interface with two main sections. On the left, under the heading "Sign in with your corporate ID", there is a teal button labeled "MYSSO". Below this, under "Sign In with your social account", there are four buttons: "Continue with Apple" (black with white Apple logo), "Continue with Login with Amazon" (yellow with Amazon logo), "Continue with Google" (blue with Google logo), and "Continue with Facebook" (dark blue with Facebook logo). A small note at the bottom of this section reads "We won't post to any of your accounts without asking first". On the right, under the heading "Sign in with your username and password", there are two input fields: "Username" and "Password". A blue "Sign in" button is positioned below these fields. A link "Forgot your password?" is located between the password field and the sign-in button. A vertical line separates the two sections, with the word "or" centered between them. At the bottom of the right section, there is a link "Need an account? Sign up".

## With only Amazon Cognito as a sign-in provider



Sign in with your username and password

Username

Password

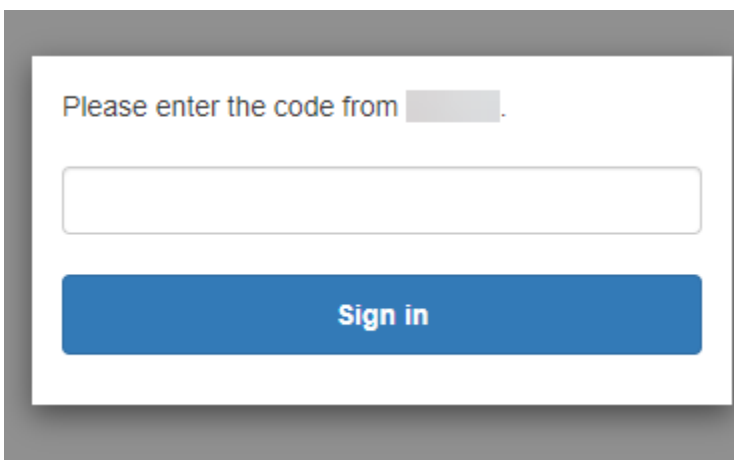
[Forgot your password?](#)

**Sign in**

Need an account? [Sign up](#)

2. Anda mungkin telah menyiapkan MFA saat mendaftar di aplikasi. Masukkan kode MFA yang Anda terima dalam pesan SMS, atau ditampilkan di aplikasi autentikator Anda. Anda harus memasukkan kode ini dalam waktu 3 menit.

## With an authenticator app

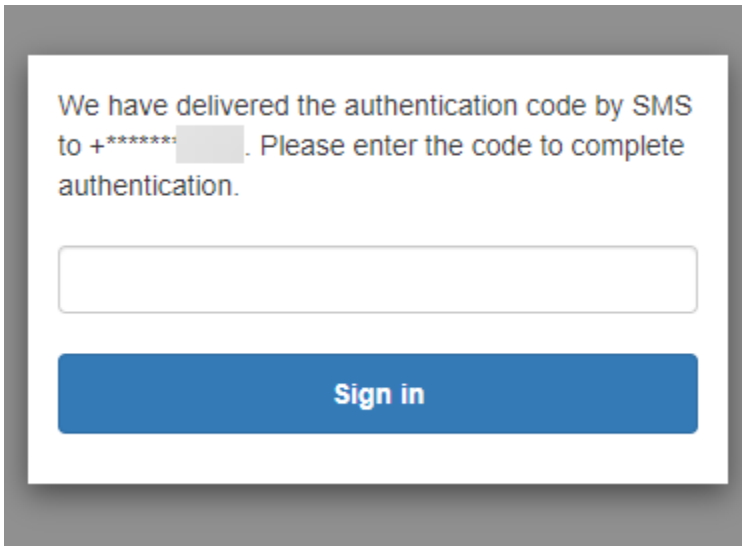


Please enter the code from  .

**Sign in**



## With an SMS code



3. Setelah Anda masuk dan menyelesaikan MFA, Amazon Cognito memberikan akses ke aplikasi Anda.

## Cara mengatur ulang kata sandi dengan UI yang dihosting Amazon Cognito

Panduan ini menunjukkan cara mengatur ulang kata sandi Anda di aplikasi yang menggunakan Amazon Cognito.

### Note

Saat masuk ke aplikasi yang menggunakan antarmuka pengguna (UI) yang dihosting Amazon Cognito, Anda mungkin melihat halaman yang telah disesuaikan oleh pemilik aplikasi di luar konfigurasi dasar yang ditunjukkan dalam panduan ini.

1. Bergantung pada opsi yang dipilih pemilik aplikasi, Anda mungkin memiliki pilihan penyedia untuk masuk, atau Anda mungkin hanya melihat prompt untuk nama pengguna dan kata sandi. Saat Anda masuk dengan nama pengguna dan sandi dari halaman ini, Amazon Cognito adalah penyedia login Anda. Jika tidak, penyedia login Anda diwakili oleh tombol yang Anda pilih.

Jika Anda biasanya memilih penyedia dari halaman login, dan kata sandi Anda tidak berfungsi, ikuti prosedur untuk mengatur ulang kata sandi Anda dengan penyedia. Jika Amazon Cognito adalah penyedia login Anda, pilih Lupa kata sandi Anda?

## With multiple sign-in providers

Sign in with your corporate ID

MYSSO

Sign In with your social account

Continue with Apple

Continue with Login with Amazon

Continue with Google

Continue with Facebook

We won't post to any of your accounts without asking first

or

Sign in with your username and password

Username

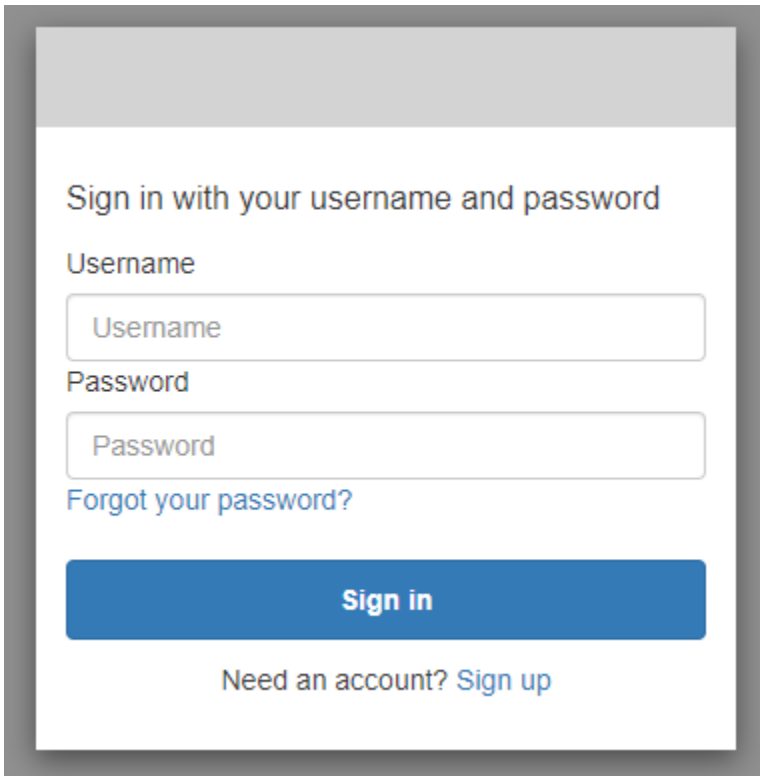
Password

Forgot your password?

Sign in

Need an account? [Sign up](#)

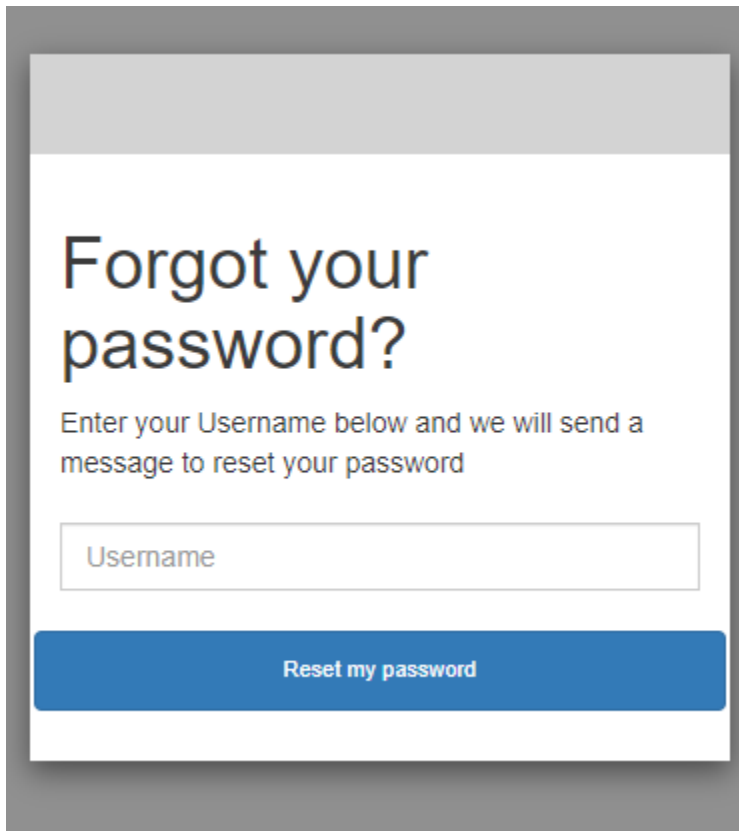
## With only Amazon Cognito as a sign-in provider



The image shows a sign-in form with the following elements:

- Header: "Sign in with your username and password"
- Label: "Username"
- Input field: "Username"
- Label: "Password"
- Input field: "Password"
- Link: "Forgot your password?"
- Button: "Sign in"
- Text: "Need an account? [Sign up](#)"

2. Pada Lupa kata sandi Anda? Halaman, Amazon Cognito meminta Anda untuk informasi yang Anda gunakan untuk masuk. Ini mungkin nama pengguna, alamat email, atau nomor telepon Anda.

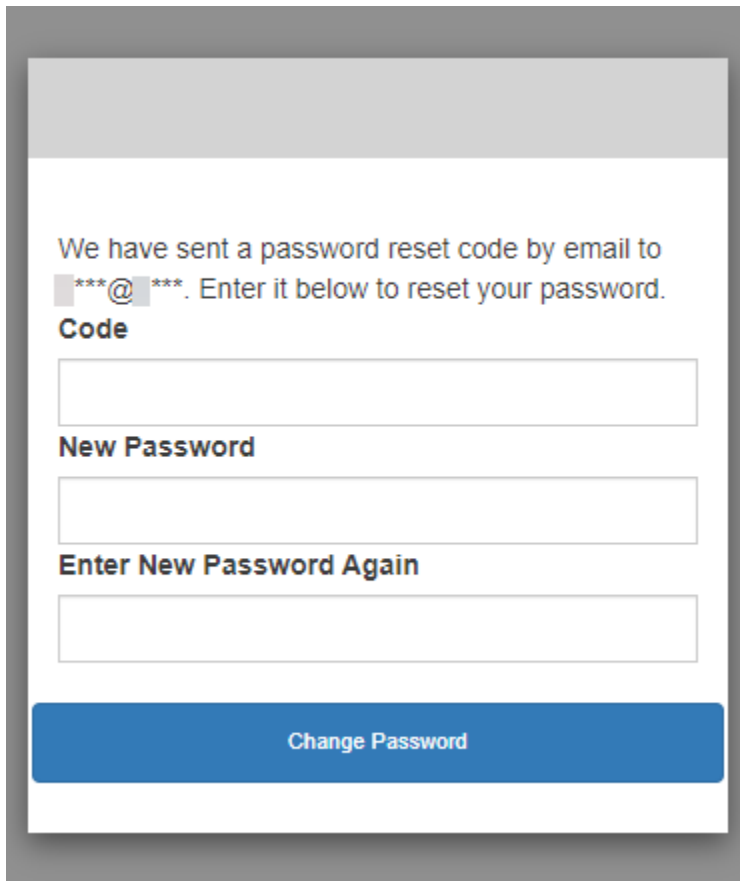


Forgot your password?

Enter your Username below and we will send a message to reset your password

3. Amazon Cognito akan mengirimkan kode kepada Anda sebagai pesan email atau sebagai pesan teks SMS.

Masukkan kode yang Anda terima, dan masukkan kata sandi baru Anda dua kali di bidang yang disediakan. Anda harus memasukkan kode reset Anda dalam waktu 8 menit.



The image shows a mobile-style form for password reset. At the top, it says "We have sent a password reset code by email to [redacted]@[redacted]. Enter it below to reset your password." Below this is a label "Code" followed by a text input field. Then a label "New Password" followed by a text input field. Below that is a label "Enter New Password Again" followed by another text input field. At the bottom is a large blue button with the text "Change Password".

4. Setelah Anda mengubah kata sandi, kembali ke halaman login dan masuk dengan kata sandi baru Anda.

## Cakupan, M2M, dan otorisasi API dengan server sumber daya

Setelah mengonfigurasi domain untuk kumpulan pengguna, Amazon Cognito secara otomatis menyediakan server otorisasi OAuth 2.0 dan UI web yang dihosting dengan halaman pendaftaran dan login yang dapat ditampilkan aplikasi kepada pengguna Anda. Untuk mengetahui informasi selengkapnya, lihat [Tambahkan klien aplikasi dengan UI yang dihosting](#). Anda dapat memilih cakupan yang Anda inginkan untuk ditambahkan oleh server otorisasi ke token akses. Cakupan mengotorisasi akses ke server sumber daya dan data pengguna.

Server sumber daya adalah server [API OAuth 2.0](#). Untuk mengamankan sumber daya yang dilindungi akses, ini memvalidasi bahwa token akses dari kumpulan pengguna Anda berisi cakupan yang mengotorisasi metode dan jalur yang diminta di API yang dilindunginya. Ini memverifikasi penerbit berdasarkan tanda tangan token, validitas berdasarkan waktu kedaluwarsa token, dan tingkat akses berdasarkan cakupan dalam klaim token. Lingkup kumpulan pengguna ada dalam

scope klaim token akses. Untuk informasi selengkapnya tentang klaim dalam token akses Amazon Cognito, lihat. [Menggunakan token akses](#)

Dengan Amazon Cognito, cakupan dalam token akses dapat mengotorisasi akses ke API eksternal atau ke atribut pengguna. Anda dapat mengeluarkan token akses ke pengguna lokal, pengguna federasi, atau identitas mesin.

## achine-to-machine Otorisasi M (M2M)

Amazon Cognito mendukung aplikasi yang mengakses data API dengan identitas mesin. Identitas mesin di kumpulan pengguna adalah [klien rahasia](#) yang berjalan di server aplikasi dan terhubung ke API jarak jauh. Operasi mereka terjadi tanpa interaksi pengguna: tugas terjadwal, aliran data, atau pembaruan aset. Ketika klien ini mengotorisasi permintaan mereka dengan token akses, mereka melakukan otorisasi machine to machine, atau M2M. Dalam otorisasi M2M, rahasia bersama menggantikan kredensi pengguna dalam kontrol akses.

Aplikasi yang mengakses API dengan otorisasi M2M harus memiliki ID klien dan rahasia klien. Di kumpulan pengguna, Anda harus membuat klien aplikasi yang mendukung hibah kredensi klien. Untuk mendukung kredensi klien, klien aplikasi Anda harus memiliki rahasia klien dan Anda harus memiliki domain kumpulan pengguna. Dalam alur ini, identitas mesin Anda meminta token akses langsung dari [Titik akhir token](#). Anda hanya dapat mengotorisasi cakupan kustom dari [server sumber daya](#) dalam token akses untuk hibah kredensi klien. Untuk informasi selengkapnya tentang menyiapkan klien aplikasi, lihat [Klien aplikasi kumpulan pengguna](#).

Token akses dari hibah kredensial klien adalah pernyataan operasi yang dapat diverifikasi yang ingin Anda izinkan identitas mesin Anda diminta dari API. Untuk mempelajari selengkapnya tentang cara token akses mengotorisasi permintaan API, lanjutkan membaca. Untuk contoh aplikasi, lihat [Amazon Cognito dan API Gateway berbasis mesin ke otorisasi mesin menggunakan CDK](#). AWS

Otorisasi M2M memiliki model penagihan yang berbeda dari cara pengguna aktif bulanan (MAUs) ditagih. Jika autentikasi pengguna membawa biaya per pengguna aktif, penagihan M2M mencerminkan kredensi klien aktif klien aplikasi dan total volume permintaan token. Untuk informasi selengkapnya, lihat [Harga Amazon Cognito](#). Untuk mengontrol biaya otorisasi M2M, optimalkan durasi token akses dan jumlah permintaan token yang dibuat aplikasi Anda. Lihat [Token cache](#) cara menggunakan caching API Gateway untuk mengurangi permintaan token baru dalam otorisasi M2M.

Untuk informasi tentang mengoptimalkan operasi Amazon Cognito yang menambahkan biaya ke tagihan AWS Anda, lihat. [Mengelola biaya](#)

## Tentang cakupan

Sebuah cakupan adalah tingkat akses yang dapat diminta aplikasi ke sumber daya. Dalam token akses Amazon Cognito, ruang lingkup didukung oleh kepercayaan yang Anda atur dengan kumpulan pengguna Anda: penerbit token akses tepercaya dengan tanda tangan digital yang dikenal.

Kumpulan pengguna dapat menghasilkan token akses dengan cakupan yang membuktikan bahwa pelanggan Anda diizinkan untuk mengelola sebagian atau semua profil pengguna mereka sendiri, atau untuk mengambil data dari API back-end. Kumpulan pengguna Amazon Cognito mengeluarkan token akses dengan kumpulan pengguna cakupan API yang dicadangkan, cakupan khusus, dan cakupan standar.

Pengguna mengumpulkan cakupan API yang dicadangkan

Ruang `aws.cognito.signin.user.admin` lingkup mengotorisasi API kumpulan pengguna Amazon Cognito. Ini memberi wewenang kepada pembawa token akses untuk menanyakan dan memperbarui semua informasi tentang pengguna kumpulan pengguna dengan, misalnya, operasi [getUser](#) dan [updateUserAttributes](#) API. Saat Anda mengautentikasi pengguna Anda dengan API kumpulan pengguna Amazon Cognito, ini adalah satu-satunya cakupan yang Anda terima dalam token akses. Ini juga satu-satunya cakupan yang Anda perlukan untuk membaca dan menulis atribut pengguna yang telah Anda otorisasi untuk dibaca dan ditulis oleh klien aplikasi Anda. Anda juga dapat meminta ruang lingkup ini dalam permintaan ke Anda [Otorisasi titik akhir](#). Cakupan ini saja tidak cukup untuk meminta atribut pengguna dari [Titik akhir UserInfo](#). Untuk token akses yang mengotorisasi API kumpulan pengguna dan `userInfo` permintaan untuk pengguna Anda, Anda harus meminta cakupan `openid` dan `aws.cognito.signin.user.admin` permintaan. `/oauth2/authorize`

Lingkup kustom

Cakupan kustom mengotorisasi permintaan ke API eksternal yang dilindungi oleh server sumber daya. Anda dapat meminta cakupan khusus dengan jenis cakupan lainnya. Anda dapat menemukan informasi lebih lanjut tentang cakupan khusus di seluruh halaman ini.

Lingkup standar

Saat Anda mengautentikasi pengguna dengan server otorisasi OAuth 2.0 kumpulan pengguna Anda, termasuk dengan UI yang dihosting, Anda harus meminta cakupan. Anda dapat mengautentikasi kumpulan pengguna pengguna lokal dan pengguna federasi pihak ketiga di server otorisasi Amazon Cognito Anda. Cakupan standar OAuth 2.0 mengotorisasi aplikasi Anda untuk membaca informasi pengguna dari kumpulan pengguna Anda [Titik akhir UserInfo](#). Model OAuth, tempat Anda melakukan

kueri atribut pengguna dari `userInfo` titik akhir, dapat mengoptimalkan aplikasi Anda untuk permintaan atribut pengguna dengan volume tinggi. `userInfo` titik akhir mengembalikan atribut pada tingkat izin yang ditentukan oleh cakupan dalam token akses. Anda dapat mengotorisasi klien aplikasi Anda untuk mengeluarkan token akses dengan cakupan OAuth 2.0 standar berikut.

## openid

Cakupan minimum untuk kueri OpenID Connect (OIDC). Mengotorisasi token ID, klaim pengenalan `uniquesub`, dan kemampuan untuk meminta cakupan lainnya.

### Note

Saat Anda meminta `openid` cakupan dan tidak ada yang lain, token dan `userInfo` respons ID kumpulan pengguna menyertakan klaim untuk semua atribut pengguna yang dapat dibaca oleh klien aplikasi Anda. Saat Anda meminta `openid` dan cakupan standar lainnya seperti `profileemail`, dan `phone`, konten token ID dan respons [User Info](#) terbatas pada batasan cakupan tambahan.

Misalnya, permintaan ke [Otorisasi titik akhir](#) with parameter `scope=openid+email` mengembalikan token ID dengan `sub`, `email`, dan `email_verified`. Token akses dari permintaan ini mengembalikan atribut yang sama dari [Titik akhir User Info](#). Permintaan dengan parameter `scope=openid` mengembalikan semua atribut yang dapat dibaca klien dalam token ID dan dari `userInfo`

## profile

Mengotorisasi semua atribut pengguna yang dapat dibaca oleh klien aplikasi.

## Email

Mengotorisasi atribut pengguna `email` dan `email_verified`. Amazon Cognito kembali `email_verified` jika memiliki nilai yang ditetapkan secara eksplisit.

## telepon

Mengotorisasi atribut pengguna `phone_number` dan `phone_number_verified`.

## Tentang server sumber daya

API server sumber daya mungkin memberikan akses ke informasi dalam database, atau mengontrol sumber daya TI Anda. Token akses Amazon Cognito dapat mengotorisasi akses ke API yang



mendukung OAuth 2.0. API REST Amazon API Gateway memiliki [dukungan bawaan](#) untuk otorisasi dengan token akses Amazon Cognito. Aplikasi Anda meneruskan token akses dalam panggilan API ke server sumber daya. Server sumber daya memeriksa token akses untuk menentukan apakah akses harus diberikan.

Amazon Cognito mungkin membuat pembaruan masa depan untuk skema token akses kumpulan pengguna. Jika aplikasi Anda menganalisis konten token akses sebelum meneruskannya ke API, Anda harus merekayasa kode Anda untuk menerima pembaruan skema.

Cakupan kustom ditentukan oleh Anda, dan memperluas kemampuan otorisasi kumpulan pengguna untuk menyertakan tujuan yang tidak terkait dengan kueri dan modifikasi pengguna dan atributnya. Misalnya, jika Anda memiliki server sumber daya untuk foto, itu mungkin menentukan dua cakupan: `photos.read` untuk akses baca ke foto dan `photos.write` untuk akses tulis/hapus. Anda dapat mengonfigurasi API untuk menerima token akses untuk otorisasi, dan memberikan HTTP GET permintaan untuk mengakses token dengan `photos.read` scope klaim, dan HTTP POST permintaan ke token dengan `photos.write`. Ini adalah cakupan khusus.

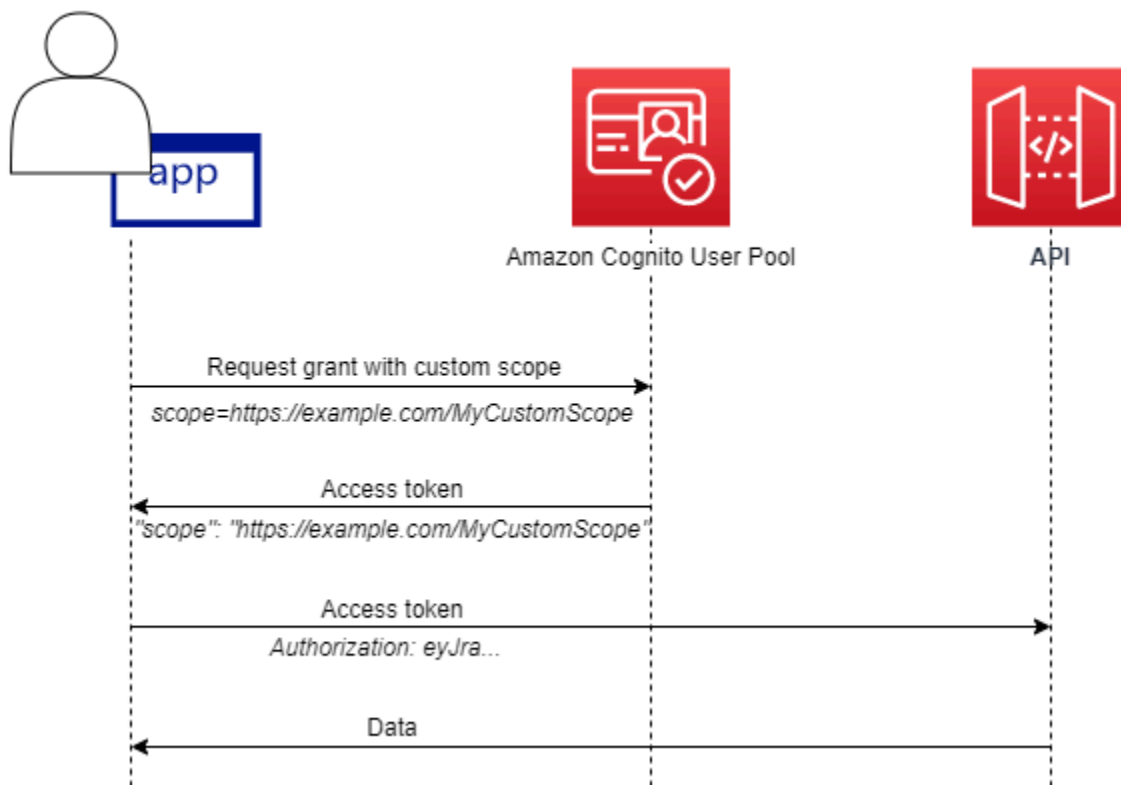
#### Note

Server sumber daya Anda harus memverifikasi tanda tangan token akses dan tanggal kedaluwarsa sebelum memproses klaim apa pun di dalam token. Untuk informasi selengkapnya tentang memverifikasi token, lihat [Memverifikasi JSON Web Token](#). Untuk informasi selengkapnya tentang memverifikasi dan menggunakan token kumpulan pengguna di Amazon API Gateway, lihat blog [Mengintegrasikan Kumpulan Pengguna Amazon Cognito dengan API Gateway](#). API Gateway adalah pilihan yang baik untuk memeriksa token akses dan melindungi sumber daya Anda. Untuk lebih lanjut tentang pengotorisasi Lambda API Gateway, lihat [Gunakan pengotorisasi Lambda API Gateway](#).

## Gambaran Umum

Dengan Amazon Cognito, Anda dapat membuat server Sumber Daya OAuth 2.0 dan mengaitkan cakupan Kustom dengannya. Cakupan kustom dalam token akses mengotorisasi tindakan tertentu di API Anda. Anda dapat mengotorisasi klien aplikasi apa pun di kumpulan pengguna untuk mengeluarkan cakupan kustom dari server sumber daya mana pun. Kaitkan cakupan kustom Anda dengan klien aplikasi dan minta cakupan tersebut dalam hibah kode otorisasi OAuth 2.0, hibah implisit, dan hibah kredensial klien dari [Titik akhir token](#) Amazon Cognito menambahkan cakupan khusus ke scope klaim dalam token akses. Klien dapat menggunakan token akses terhadap server

sumber dayanya, yang membuat keputusan otorisasi berdasarkan cakupan yang ada dalam token. Untuk informasi selengkapnya tentang cakupan token akses, lihat [Menggunakan Token dengan Kolam Pengguna](#).



Untuk mendapatkan token akses dengan cakupan kustom, aplikasi Anda harus membuat permintaan [Titik akhir token](#) untuk menukarkan kode otorisasi atau meminta pemberian kredensi klien. Di UI yang dihosting, Anda juga dapat meminta cakupan khusus dalam token akses dari hibah implisit.

#### **Note**

Karena mereka dirancang untuk otentikasi interaktif manusia dengan kumpulan pengguna sebagai IDP, [InitiateAuth](#) dan [AdminInitiateAuth](#) permintaan hanya menghasilkan scope klaim dalam token akses dengan nilai tunggal. `aws.cognito.signin.user.admin`

## Mengelola Server Sumber Daya dan Lingkup Kustom

Saat membuat server sumber daya, Anda harus memberikan nama server sumber daya dan pengenal server sumber daya. Untuk setiap lingkup yang Anda buat di server sumber daya, Anda harus memberikan nama lingkup dan deskripsi.

- Nama server sumber daya: Nama yang ramah untuk server sumber daya, seperti `Solar system object tracker` atau `Photo API`.
- Pengidentifikasi server sumber daya: Pengidentifikasi unik untuk server sumber daya. Pengenal adalah nama apa pun yang ingin Anda kaitkan dengan API Anda, misalnya `solar-system-data`. Anda dapat mengonfigurasi pengidentifikasi yang lebih panjang `https://solar-system-data-api.example.com` seperti referensi yang lebih langsung ke jalur URI API, tetapi string yang lebih panjang meningkatkan ukuran token akses.
- Nama lingkup: Nilai yang Anda inginkan dalam scope klaim Anda. Misalnya, `sunproximity.read`.
- Deskripsi: Deskripsi ruang lingkup yang ramah. Misalnya, `Check current proximity to sun`.

Amazon Cognito dapat menyertakan cakupan khusus dalam token akses untuk pengguna mana pun, baik lokal untuk kumpulan pengguna Anda atau digabungkan dengan penyedia identitas pihak ketiga. Anda dapat memilih cakupan untuk token akses pengguna Anda selama alur otentikasi dengan server otorisasi OAuth 2.0 yang menyertakan UI yang dihosting. Otentikasi pengguna Anda harus dimulai dari [Otorisasi titik akhir](#) dengan scope sebagai salah satu parameter permintaan. Berikut ini adalah format yang direkomendasikan untuk server sumber daya. Untuk pengenal, gunakan nama yang ramah API. Untuk cakupan kustom, gunakan tindakan yang mereka otorisasi.

```
resourceServerIdentifier/scopeName
```

Misalnya, Anda telah menemukan asteroid baru di sabuk Kuiper dan Anda ingin mendaftarkannya melalui API `AndaSolar-system-data`. Ruang lingkup yang mengotorisasi operasi tulis ke database asteroid adalah `asteroids.add`. Ketika Anda meminta token akses yang akan mengizinkan Anda untuk mendaftarkan penemuan Anda, format parameter permintaan scope HTTPS Anda sebagai `scope=solar-system-data/asteroids.add`.

Menghapus ruang lingkup dari server sumber daya tidak menghapus hubungannya dengan semua klien. Sebaliknya, ruang lingkup ditandai tidak aktif. Amazon Cognito tidak menambahkan cakupan tidak aktif untuk mengakses token, tetapi sebaliknya berjalan seperti biasa jika aplikasi Anda memintanya. Jika Anda menambahkan cakupan ke server sumber daya Anda lagi nanti, maka Amazon Cognito kembali menuliskannya ke token akses. Jika Anda meminta cakupan yang belum dikaitkan dengan klien aplikasi, terlepas dari apakah Anda menghapusnya dari server sumber daya kumpulan pengguna, autentikasi gagal.

Anda dapat menggunakan API AWS Management Console, atau CLI untuk menentukan server sumber daya dan cakupan untuk kumpulan pengguna Anda.

## Mendefinisikan server sumber daya untuk kumpulan pengguna Anda (AWS Management Console)

Anda dapat menggunakan AWS Management Console untuk menentukan server sumber daya untuk kumpulan pengguna Anda.

Cara menentukan server sumber daya

1. Masuk ke [konsol Amazon Cognito](#).
2. Di panel navigasi, pilih Kumpulan Pengguna, dan pilih kumpulan pengguna yang ingin Anda edit.
3. Pilih tab Integrasi aplikasi dan temukan Server sumber daya.
4. Pilih Buat server sumber daya.
5. Masukkan nama server Sumber Daya. Misalnya, Photo Server.
6. Masukkan pengenalan server Sumber Daya. Misalnya, `com.example.photos`.
7. Masukkan cakupan khusus untuk sumber daya Anda, seperti `read` dan `write`.
8. Untuk setiap nama Lingkup, masukkan Deskripsi, seperti `view your photos` dan `update your photos`.
9. Pilih Buat.

Cakupan kustom Anda dapat ditinjau di tab Integrasi aplikasi di bawah Server sumber daya, di kolom Cakupan kustom. Cakupan khusus dapat diaktifkan untuk klien aplikasi dari tab Integrasi aplikasi di bawah Klien aplikasi. Pilih klien aplikasi, cari Setelan UI yang Dihosting, dan pilih Edit. Tambahkan cakupan kustom dan pilih Simpan perubahan.

## Mendefinisikan server sumber daya untuk kumpulan pengguna Anda (AWS CLI dan AWS API)

Gunakan perintah berikut untuk menentukan pengaturan server sumber daya untuk kolam pengguna Anda.

Cara membuat server sumber daya

- AWS CLI: `aws cognito-idp create-resource-server`
- AWS API: [CreateResourceServer](#)

Cara mendapatkan informasi tentang pengaturan server sumber daya Anda

- AWS CLI: `aws cognito-idp describe-resource-server`
- AWS API: [DescribeResourceServer](#)

Cara mendaftarkan informasi tentang semua server sumber daya untuk kolam pengguna

- AWS CLI: `aws cognito-idp list-resource-servers`
- AWS API: [ListResourceServers](#)

Cara menghapus server sumber daya

- AWS CLI: `aws cognito-idp delete-resource-server`
- AWS API: [DeleteResourceServer](#)

Cara memperbarui pengaturan pada server sumber daya

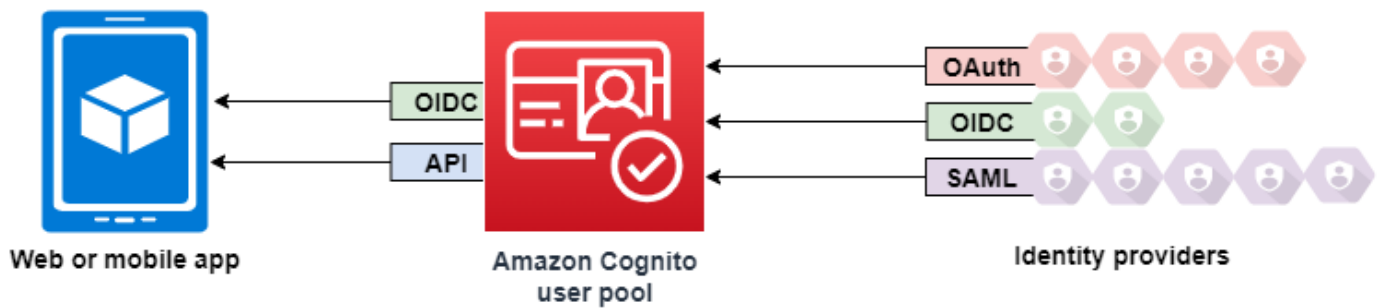
- AWS CLI: `aws cognito-idp update-resource-server`
- AWS API: [UpdateResourceServer](#)

## Menambahkan login kumpulan pengguna melalui pihak ketiga

Pengguna aplikasi Anda dapat masuk langsung melalui kumpulan pengguna, atau mereka dapat melakukan federasi melalui penyedia identitas pihak ketiga (iDP). Kumpulan pengguna mengelola overhead penanganan token yang dikembalikan dari login sosial melalui Facebook, Google, Amazon, dan Apple, dan dari OpenID Connect (OIDC) dan SAMP. IdPs Dengan UI web host bawaan, Amazon Cognito menyediakan penanganan dan manajemen token untuk pengguna yang diautentikasi dari semua. IdPs Dengan cara ini, sistem backend Anda dapat melakukan standarisasi pada satu set token kumpulan pengguna.

## Cara kerja masuk federasi di kumpulan pengguna Amazon Cognito

Masuk melalui pihak ketiga (federasi) tersedia di kolam pengguna Amazon Cognito. Fitur ini independen dari federasi melalui kolam identitas Amazon Cognito (identitas federasi).



Amazon Cognito adalah direktori pengguna dan penyedia identitas OAuth 2.0 (iDP). Saat Anda memasukkan pengguna lokal ke direktori Amazon Cognito, kumpulan pengguna Anda adalah iDP ke aplikasi Anda. Pengguna lokal ada secara eksklusif di direktori kumpulan pengguna Anda tanpa federasi melalui iDP eksternal.

Saat Anda menghubungkan Amazon Cognito ke social, SAMP, atau OpenID Connect (OIDC IdPs), kumpulan pengguna akan berfungsi sebagai jembatan antara beberapa penyedia layanan dan aplikasi Anda. Untuk IDP Anda, Amazon Cognito adalah penyedia layanan (SP). Anda IdPs meneruskan token ID OIDC atau pernyataan SAMP ke Amazon Cognito. Amazon Cognito membaca klaim tentang pengguna Anda di token atau pernyataan dan memetakan klaim tersebut ke profil pengguna baru di direktori kumpulan pengguna Anda.

Amazon Cognito kemudian membuat profil pengguna untuk pengguna federasi Anda di direktorinya sendiri. Amazon Cognito menambahkan atribut ke pengguna Anda berdasarkan klaim dari IDP Anda dan, dalam kasus OIDC dan penyedia identitas sosial, titik akhir publik yang dioperasikan IDP. `userInfo` Atribut pengguna Anda berubah di kumpulan pengguna Anda saat atribut IDP yang dipetakan berubah. Anda juga dapat menambahkan lebih banyak atribut independen dari atribut dari iDP.

Setelah Amazon Cognito membuat profil untuk pengguna federasi Anda, ia mengubah fungsinya dan menampilkan dirinya sebagai iDP ke aplikasi Anda, yang sekarang menjadi SP. Amazon Cognito adalah kombinasi OIDC dan OAuth 2.0 iDP. Ini menghasilkan token akses, token ID, dan token penyegaran. Untuk informasi lebih lanjut tentang token, lihat [Menggunakan token dengan kolam pengguna](#).

Anda harus mendesain aplikasi yang terintegrasi dengan Amazon Cognito untuk mengautentikasi dan mengotorisasi pengguna Anda, baik federasi maupun lokal.

# Tanggung jawab aplikasi sebagai penyedia layanan dengan Amazon Cognito

## Verifikasi dan proses informasi dalam token

Dalam sebagian besar skenario, Amazon Cognito mengalihkan pengguna yang diautentikasi ke URL aplikasi yang ditambahkan dengan kode otorisasi. Aplikasi Anda [menukar kode](#) untuk token akses, ID, dan penyegaran. Kemudian, ia harus [memeriksa validitas token](#) dan menyajikan informasi kepada pengguna Anda berdasarkan klaim dalam token.

## Menanggapi peristiwa otentikasi dengan permintaan Amazon Cognito API

Aplikasi Anda harus terintegrasi dengan [API kumpulan pengguna Amazon Cognito dan titik akhir API autentikasi](#). API otentikasi menandatangani pengguna Anda masuk dan keluar, dan mengelola token. API kumpulan pengguna memiliki berbagai operasi yang mengelola kumpulan pengguna, pengguna, dan keamanan lingkungan otentikasi Anda. Aplikasi Anda harus tahu apa yang harus dilakukan selanjutnya ketika menerima respons dari Amazon Cognito.

## Hal-hal yang perlu diketahui tentang kumpulan pengguna Amazon Cognito login pihak ketiga

- Jika Anda ingin pengguna Anda masuk dengan penyedia federasi, Anda harus memilih domain. Ini mengatur UI yang dihosting Amazon Cognito dan UI yang [dihosting dan titik akhir OIDC](#). Untuk informasi selengkapnya, lihat [Menggunakan domain Anda sendiri untuk UI yang dihosting](#).
- Anda tidak dapat masuk ke pengguna federasi dengan operasi API seperti [InitiateAuth](#) dan [AdminInitiateAuth](#). Pengguna federasi hanya dapat masuk dengan [Titik akhir masuk](#) atau [Otorisasi titik akhir](#).
- [Otorisasi titik akhir](#) ini adalah titik akhir pengalihan. Jika Anda memberikan `identity_provider` parameter `idp_identifier` atau dalam permintaan Anda, parameter tersebut akan dialihkan secara diam-diam ke IDP Anda, melewati UI yang dihosting. Jika tidak, itu dialihkan ke UI yang dihosting. [Titik akhir masuk](#) Sebagai contoh, lihat [Contoh skenario: bookmark aplikasi Amazon Cognito di dasbor perusahaan](#).
- Saat UI yang dihosting mengalihkan sesi ke iDP federasi, Amazon Cognito menyertakan header dalam permintaan `user-agent`. `Amazon/Cognito`
- Amazon Cognito memperoleh `username` atribut untuk profil pengguna federasi dari kombinasi pengenal tetap dan nama iDP Anda. Untuk menghasilkan nama pengguna yang sesuai dengan

persyaratan kustom Anda, buat pemetaan ke `preferred_username` atribut. Untuk informasi selengkapnya, lihat [Hal yang perlu diketahui tentang pemetaan](#).

Contoh: `MyIDP_bob@example.com`

- Amazon Cognito mencatat informasi tentang identitas pengguna federasi Anda ke atribut, dan klaim dalam token ID, yang dipanggil. `identities` Klaim ini berisi penyedia pengguna Anda dan ID unik mereka dari penyedia. Anda tidak dapat mengubah `identities` atribut di profil pengguna secara langsung. Untuk informasi selengkapnya tentang cara menautkan pengguna federasi, lihat [Menautkan pengguna gabungan ke profil pengguna yang ada](#).
- Saat memperbarui IDP dalam permintaan [UpdateIdentityProvider](#) API, perubahan dapat memakan waktu hingga satu menit untuk muncul di UI yang dihosting.
- Amazon Cognito mendukung hingga 20 pengalihan HTTP antara dirinya dan IDP Anda.
- Saat pengguna Anda masuk dengan UI yang dihosting, browser mereka menyimpan cookie sesi masuk terenkripsi yang mencatat klien dan penyedia tempat mereka masuk. Jika mereka mencoba masuk lagi dengan parameter yang sama, UI yang dihosting menggunakan kembali sesi yang ada yang belum kedaluwarsa, dan pengguna mengautentikasi tanpa memberikan kredensial lagi. Jika pengguna Anda masuk lagi dengan IDP yang berbeda, termasuk peralihan ke atau dari login kumpulan pengguna lokal, mereka harus memberikan kredensial dan membuat sesi login baru.

Anda dapat menetapkan kumpulan pengguna apa pun IdPs ke klien aplikasi apa pun, dan pengguna hanya dapat masuk dengan iDP yang Anda tetapkan ke klien aplikasi mereka.

## Topik

- [Mengkonfigurasi penyedia identitas untuk kumpulan pengguna Anda](#)
- [Menggunakan penyedia identitas sosial dengan kumpulan pengguna](#)
- [Menggunakan penyedia identitas SAMP dengan kumpulan pengguna](#)
- [Menggunakan penyedia identitas OIDC dengan kumpulan pengguna](#)
- [Menentukan pemetaan atribut penyedia identitas untuk kumpulan pengguna Anda](#)
- [Menautkan pengguna gabungan ke profil pengguna yang ada](#)

## Mengkonfigurasi penyedia identitas untuk kumpulan pengguna Anda

Di tab Pengalaman masuk di bawah login penyedia identitas gabungan, Anda dapat menambahkan penyedia identitas (IdPs) ke kumpulan pengguna Anda. Untuk informasi selengkapnya, lihat [Menambahkan login kumpulan pengguna melalui pihak ketiga](#).



## Topik

- [Mengatur login pengguna dengan iDP sosial](#)
- [Mengatur login pengguna dengan OIDC iDP](#)
- [Mengatur login pengguna dengan SALL iDP](#)

## Mengatur login pengguna dengan iDP sosial

Anda dapat menggunakan federasi untuk mengintegrasikan kumpulan pengguna Amazon Cognito dengan penyedia identitas sosial seperti Facebook, Google, dan Login with Amazon.

Untuk menambahkan penyedia identitas sosial, Anda terlebih dahulu membuat akun developer dengan penyedia identitas. Setelah Anda memiliki akun pengembang, daftarkan aplikasi Anda ke penyedia identitas. Penyedia identitas membuat ID aplikasi dan rahasia aplikasi untuk aplikasi Anda, dan Anda mengonfigurasi nilai-nilai tersebut di kolam pengguna Amazon Cognito Anda.

- [Platform identitas Google](#)
- [Facebook untuk pengembang](#)
- [Login with Amazon](#)
- [Masuk dengan Apple](#)

Untuk mengintegrasikan login pengguna dengan iDP sosial

1. Masuk ke [konsol Amazon Cognito](#). Jika diminta, masukkan AWS kredensial Anda.
2. Di panel navigasi, pilih Kumpulan Pengguna, dan pilih kumpulan pengguna yang ingin Anda edit.
3. Pilih tab Pengalaman masuk dan temukan Masuk Federasi.
4. Pilih Tambahkan penyedia identitas, atau pilih penyedia identitas Facebook, Google, Amazon, atau Apple yang telah Anda konfigurasi, cari informasi penyedia Identitas, dan pilih Edit. Untuk informasi selengkapnya tentang menambahkan penyedia identitas sosial, lihat [Menggunakan penyedia identitas sosial dengan kumpulan pengguna](#).
5. Masukkan informasi penyedia identitas sosial Anda dengan menyelesaikan salah satu langkah berikut, berdasarkan pilihan IDP Anda:

Facebook, Google, dan Login with Amazon

Masukkan ID aplikasi dan rahasia aplikasi yang Anda terima saat membuat aplikasi klien.

## Masuk dengan Apple

Masukkan ID layanan yang Anda berikan ke Apple, dan ID tim, ID kunci, dan kunci pribadi yang Anda terima saat membuat klien aplikasi.

- Untuk cakupan Resmi, masukkan nama cakupan penyedia identitas sosial yang ingin Anda petakan ke atribut kumpulan pengguna. Cakupan menentukan atribut pengguna, seperti nama dan email, yang ingin Anda akses dengan aplikasi Anda. Saat memasukkan cakupan, gunakan panduan berikut berdasarkan pilihan IDP Anda:

- Facebook — Pisahkan cakupan dengan koma. Sebagai contoh:

```
public_profile, email
```

- Google, Login with Amazon, dan Masuk dengan Apple — Pisahkan cakupan dengan spasi. Sebagai contoh:

- Google: `profile email openid`
- Login with Amazon: `profile postal_code`
- Masuk dengan Apple: `name email`

### Note

Untuk Masuk dengan Apple (konsol), gunakan kotak centang untuk memilih cakupan.

- Pilih Simpan perubahan.
- Dari tab Integrasi klien aplikasi, pilih salah satu klien Aplikasi dalam daftar, lalu pilih Edit pengaturan UI yang dihosting. Tambahkan penyedia identitas sosial baru ke klien aplikasi di bawah Penyedia identitas.
- Pilih Simpan perubahan.

Untuk informasi lebih lanjut tentang sosial IdPs, lihat [Menggunakan penyedia identitas sosial dengan kumpulan pengguna](#).

## Mengatur login pengguna dengan OIDC iDP

Anda dapat mengintegrasikan login pengguna dengan penyedia identitas OpenID Connect (OIDC) (IDP) seperti Salesforce atau Ping Identity.

Untuk menambahkan penyedia OIDC ke kumpulan pengguna

1. Masuk ke [Konsol Amazon Cognito](#). Jika diminta, masukkan AWS kredensial Anda.
2. Pilih User Pools dari menu navigasi.
3. Pilih kolam pengguna yang ada dari daftar, atau [buat kolam pengguna](#).
4. Pilih tab Pengalaman masuk. Cari Masuk Federasi dan pilih Tambahkan penyedia identitas.
5. Pilih penyedia identitas OpenID Connect.
6. Masukkan nama unik ke dalam nama Penyedia.
7. Masukkan ID klien yang Anda terima dari penyedia Anda ke ID Klien.
8. Masukkan rahasia klien yang Anda terima dari penyedia Anda ke dalam rahasia Klien.
9. Masukkan cakupan resmi untuk penyedia ini. Cakupan menentukan grup atribut pengguna mana (seperti `name danemail`) yang akan diminta aplikasi Anda dari penyedia Anda. Lingkup harus dipisahkan oleh spasi, mengikuti spesifikasi [OAuth 2.0](#).

Pengguna Anda harus menyetujui untuk memberikan atribut ini ke aplikasi Anda.

10. Pilih metode permintaan Atribut untuk menyediakan Amazon Cognito dengan metode HTTP (baik GET atau POST) yang digunakan Amazon Cognito untuk mengambil detail pengguna dari titik akhir UserInfo yang dioperasikan oleh penyedia Anda.
11. Pilih metode Pengaturan untuk mengambil titik akhir OpenID Connect baik dengan pengisian Otomatis melalui URL penerbit atau input Manual. Gunakan Isi otomatis melalui URL penerbit saat penyedia Anda memiliki `.well-known/openid-configuration` titik akhir publik tempat Amazon Cognito dapat mengambil URL,,, dan titik akhir `authorization.token userInfo jwks_uri`
12. Masukkan URL penerbit atau `authorization.token userInfo`, dan URL `jwks_uri` endpoint dari IDP Anda.

#### Note

Anda hanya dapat menggunakan nomor port 443 dan 80 dengan URL penemuan, diisi otomatis, dan dimasukkan secara manual. Login pengguna gagal jika penyedia OIDC Anda menggunakan port TCP yang tidak standar.

URL penerbit harus dimulai dengan `https://`, dan tidak boleh diakhiri dengan `/` karakter. Misalnya, Salesforce menggunakan URL ini:

```
https://login.salesforce.com
```

openid-configurationDokumen yang terkait dengan URL penerbit Anda harus menyediakan URL HTTPS untuk nilai berikut: `authorization_endpoint`, `token_endpoint`, `userinfo_endpoint`, dan  `JWKS_URI` . Demikian pula, ketika Anda memilih input Manual, Anda hanya dapat memasukkan URL HTTPS.

13. Sub klaim OIDC dipetakan ke atribut kumpulan pengguna Username secara default. Anda dapat memetakan [klaim](#) OIDC lainnya ke atribut kolam pengguna. Masukkan klaim OIDC, dan pilih atribut kumpulan pengguna yang sesuai dari daftar drop-down. Misalnya, email klaim sering dipetakan ke atribut kolam pengguna Email.
14. Petakan atribut tambahan dari penyedia identitas Anda ke kumpulan pengguna Anda. Untuk informasi selengkapnya, lihat [Menentukan pemetaan atribut Penyedia Identitas untuk kumpulan pengguna Anda](#).
15. Pilih Buat.
16. Dari tab Integrasi klien aplikasi, pilih salah satu Klien aplikasi dalam daftar dan Edit pengaturan UI yang dihosting. Tambahkan penyedia identitas OIDC baru ke klien aplikasi di bawah Penyedia identitas.
17. Pilih Simpan perubahan.

Untuk informasi lebih lanjut tentang OIDC IdPs, lihat [Menggunakan penyedia identitas OIDC dengan kumpulan pengguna](#)


## Mengatur login pengguna dengan SALL iDP

Anda dapat menggunakan federasi untuk kolam pengguna Amazon Cognito untuk berintegrasi dengan penyedia identitas (IdP) SAML. Anda menyediakan dokumen metadata, baik dengan mengunggah file atau dengan memasukkan URL titik akhir dokumen metadata. Untuk informasi tentang mendapatkan dokumen metadata untuk IdPs SAMP pihak ketiga, lihat [Mengonfigurasi penyedia identitas SAMP pihak ketiga Anda](#)

Untuk mengonfigurasi penyedia identitas SAML 2.0 di kolam pengguna Anda


1. Masuk ke [Konsol Amazon Cognito](#). Jika diminta, masukkan AWS kredensial Anda.
2. Pilih Kolam Pengguna.
3. Pilih kolam pengguna yang ada dari daftar, atau [buat kolam pengguna](#).
4. Pilih tab Pengalaman masuk. Cari Masuk Federasi dan pilih Tambahkan penyedia identitas.
5. Pilih penyedia identitas SAMP.

6. Masukkan Identifier dipisahkan dengan koma. Pengidentifikasi mengarahkan Amazon Cognito untuk memeriksa alamat email masuk pengguna, lalu mengarahkan pengguna ke penyedia yang sesuai dengan domain mereka.
7. Pilih Tambahkan alur keluar jika Anda ingin Amazon Cognito mengirim permintaan keluar yang ditandatangani ke penyedia Anda saat pengguna keluar. Konfigurasi penyedia identitas SAMP 2.0 Anda untuk mengirim respons keluar ke titik <https://mydomain.us-east-1.amazoncognito.com/saml2/logout> akhir yang dibuat Amazon Cognito saat Anda mengonfigurasi UI yang dihosting. `saml2/logout` Titik akhir menggunakan pengikatan POST.

 Note

Jika Anda memilih opsi ini dan penyedia identitas SAMP mengharapkan permintaan logout yang ditandatangani, Anda juga harus mengonfigurasi sertifikat penandatanganan yang disediakan oleh Amazon Cognito dengan IDP SALL Anda. IDP SAML akan memproses permintaan logout yang ditandatangani dan me-logout pengguna Anda dari sesi Amazon Cognito.

8. Pilih sumber dokumen Metadata. Jika penyedia identitas Anda menawarkan metadata SAMP di URL publik, Anda dapat memilih URL dokumen Metadata dan memasukkan URL publik tersebut. Jika tidak, pilih Unggah dokumen metadata dan pilih file metadata yang Anda unduh dari penyedia Anda sebelumnya.

 Note

Jika penyedia Anda memiliki titik akhir publik, sebaiknya Anda memasukkan URL dokumen metadata, daripada mengunggah file. Jika Anda menggunakan URL, Amazon Cognito menyegarkan metadata secara otomatis. Biasanya, penyegaran metadata terjadi setiap 6 jam atau sebelum metadata kedaluwarsa, mana yang lebih awal.

9. Petakan atribut antara penyedia SAMP dan aplikasi Anda untuk memetakan atribut penyedia SAMP ke profil pengguna di kumpulan pengguna Anda. Sertakan atribut wajib kumpulan pengguna Anda di peta atribut Anda.

Misalnya, ketika Anda memilih atribut `User pool email`, masukkan nama atribut SAMP seperti yang muncul dalam pernyataan SAMP dari penyedia identitas Anda. Penyedia identitas Anda mungkin menawarkan contoh pernyataan SAMP untuk referensi. Beberapa penyedia identitas

menggunakan nama sederhana, seperti `email`, sementara yang lain menggunakan nama atribut berformat URL yang mirip dengan:

```
http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress
```

## 10. Pilih Buat.

### Note

Jika Anda melihat `InvalidParameterException` saat membuat IDP SAMP dengan URL titik akhir metadata HTTPS, pastikan bahwa titik akhir metadata memiliki SSL yang diatur dengan benar dan ada sertifikat SSL yang valid yang terkait dengannya. <metadata endpoint>Salah satu contoh pengecualian tersebut adalah “Kesalahan mengambil metadata dari”.

Untuk menyiapkan IdP SAML untuk menambahkan sertifikat penandatanganan

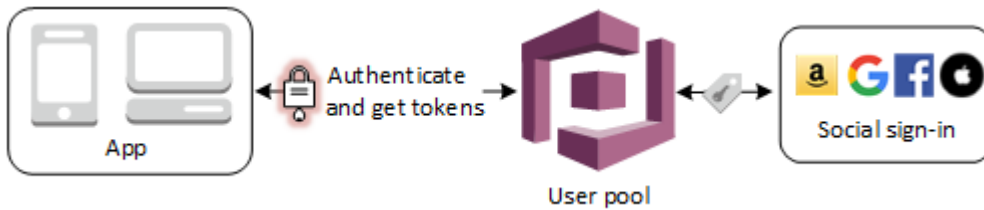
- Untuk mendapatkan sertifikat yang berisi kunci publik yang digunakan idP untuk memverifikasi permintaan logout yang ditandatangani, pilih Tampilkan sertifikat penandatanganan di bawah Penyedia SAM Aktif pada dialog SAMP di bawah Penyedia identitas di halaman konsol Federasi.

Untuk informasi lebih lanjut tentang SAMP IdPs lihat [Menggunakan penyedia identitas SAMP dengan kumpulan pengguna](#).

## Menggunakan penyedia identitas sosial dengan kumpulan pengguna

Pengguna web dan aplikasi seluler Anda dapat masuk melalui penyedia identitas (IdP) sosial seperti Facebook, Google, Amazon, dan Apple. Dengan UI web host bawaan, Amazon Cognito menyediakan penanganan dan manajemen token untuk semua pengguna yang diautentikasi. Dengan cara ini, sistem backend Anda dapat melakukan standarisasi pada satu set token kumpulan pengguna. Anda harus mengaktifkan UI yang dihosting untuk berintegrasi dengan penyedia identitas sosial yang didukung. Saat Amazon Cognito membangun UI yang Anda hosting, Amazon Cognito akan membuat titik akhir OAuth 2.0 yang Amazon Cognito dan OIDC serta penggunaan sosial Anda untuk bertukar informasi. IdPs Untuk informasi selengkapnya, lihat referensi [API Auth kumpulan pengguna Amazon Cognito](#).

Anda dapat menambahkan iDP sosial di AWS Management Console, atau Anda dapat menggunakan CLI atau AWS Amazon Cognito API.



### Note

Masuk melalui pihak ketiga (federasi) tersedia di kolam pengguna Amazon Cognito. Fitur ini independen dari federasi melalui kolam identitas Amazon Cognito (identitas federasi).

## Topik

- [Prasyarat](#)
- [Langkah 1: Daftar dengan iDP sosial](#)
- [Langkah 2: Tambahkan iDP sosial ke kolam pengguna Anda](#)
- [Langkah 3: Uji konfigurasi iDP sosial Anda](#)

## Prasyarat

Sebelum memulai, Anda perlu melakukan hal berikut:

- Kumpulan pengguna dengan klien aplikasi dan domain kumpulan pengguna. Untuk informasi selengkapnya, lihat [Membuat kolam pengguna](#).
- Sebuah iDP sosial.

## Langkah 1: Daftar dengan iDP sosial

Sebelum Anda membuat IdP sosial dengan Amazon Cognito, Anda harus mendaftarkan aplikasi Anda dengan IdP sosial untuk menerima ID klien dan rahasia klien.

Untuk mendaftarkan aplikasi dengan Facebook

1. Buat sebuah [akun developer dengan Facebook](#).

2. [Masuk](#) dengan kredensial Facebook Anda.
3. Dari menu Aplikasi Saya, pilih Buat Aplikasi Baru.
4. Masukkan nama untuk aplikasi Facebook Anda, lalu pilih Buat ID Aplikasi.
5. Di bilah navigasi kiri, pilih Pengaturan, dan kemudian Dasar.
6. Perhatikan ID Aplikasi dan Rahasia Aplikasi. Anda akan menggunakannya di bagian selanjutnya.
7. Pilih + Tambahkan Platform dari bagian bawah halaman.
8. Pilih Situs Web.
9. Di bawah Situs Web, masukkan jalur ke halaman login untuk aplikasi Anda ke URL Situs.

```
https://mydomain.us-east-1.amazoncognito.com/login?  
response_type=code&client_id=example23456789&redirect_uri=https://www.example.com
```

10. Pilih Simpan perubahan.
11. Masukkan jalur ke root domain kumpulan pengguna Anda ke Domain Aplikasi.

```
https://mydomain.us-east-1.amazoncognito.com
```

12. Pilih Simpan perubahan.
13. Dari bilah navigasi pilih Tambah Produk dan pilih Siapkan untuk produk Login Facebook.
14. Dari bilah navigasi pilih Facebook Login dan kemudian Pengaturan.

Masukkan jalur ke /oauth2/idpresponse titik akhir untuk domain kumpulan pengguna Anda ke URI Pengalihan OAuth yang Valid.

```
https://mydomain.us-east-1.amazoncognito.com/oauth2/idpresponse
```

15. Pilih Simpan perubahan.

Untuk mendaftarkan aplikasi dengan Amazon

1. Buat sebuah [akun developer dengan Amazon](#).
2. [Masuk](#) dengan kredensial Amazon Anda.
3. Anda perlu untuk membuat profil keamanan Amazon untuk menerima ID klien dan rahasia klien Amazon.



Pilih Aplikasi dan Layanan dari bilah navigasi di bagian atas halaman, lalu pilih Login with Amazon.

4. Pilih Buat Profil Keamanan.
5. Masukkan Nama Profil Keamanan, Deskripsi Profil Keamanan, dan URL Pemberitahuan Privasi Persetujuan.
6. Pilih Simpan.
7. Pilih ID Klien dan Rahasia Klien untuk menampilkan ID dan rahasia klien. Anda akan menggunakannya di bagian selanjutnya.
8. Arahkan kursor ke ikon roda gigi dan pilih Pengaturan Web, lalu pilih Edit.
9. Masukkan domain kumpulan pengguna Anda ke Asal yang Diizinkan.

```
https://mydomain.us-east-1.amazoncognito.com
```

10. Masukkan domain pool pengguna Anda dengan /oauth2/idpresponse titik akhir ke URL Pengembalian yang Diizinkan.

```
https://mydomain.us-east-1.amazoncognito.com/oauth2/idpresponse
```

11. Pilih Simpan.

Untuk mendaftarkan aplikasi dengan Google

Untuk informasi selengkapnya tentang OAuth 2.0 di platform Google Cloud, lihat [Pelajari tentang autentikasi & otorisasi](#) di dokumentasi Google Workspace for Developers.

1. Buat sebuah [akun developer dengan Google](#).
2. Masuk ke [konsol Google Cloud Platform](#).
3. Dari bilah navigasi atas, pilih Pilih proyek. Jika Anda sudah memiliki proyek di platform Google, menu ini menampilkan proyek default Anda sebagai gantinya.
4. Pilih PROYEK BARU.
5. Masukkan nama untuk produk Anda dan kemudian pilih CREATE.
6. Di bilah navigasi kiri, pilih API dan Layanan, lalu layar persetujuan OAuth.
7. Masukkan informasi Aplikasi, domain Aplikasi, domain Resmi, dan informasi kontak Pengembang. Domain resmi Anda harus menyertakan amazoncognito.com dan akar domain kustom Anda, misalnyaexample.com. Pilih SIMPAN DAN LANJUTKAN.

8. 1. Di bawah Cakupan, pilih Tambah atau hapus cakupan, dan pilih, minimal, cakupan OAuth berikut.
  1. .../auth/userinfo.email
  2. .../auth/userinfo.profile
  3. openid
9. Di bawah Uji pengguna, pilih Tambahkan pengguna. Masukkan alamat email Anda dan pengguna uji resmi lainnya, lalu pilih SIMPAN DAN LANJUTKAN.
10. Perluas bilah navigasi kiri lagi, lalu pilih API dan Layanan, lalu Kredensial.
11. Pilih CREATE CREDENTIALS, lalu OAuth client ID.
12. Pilih jenis Aplikasi dan beri nama klien Anda.
13. Di bawah JavaScript Asal resmi, pilih TAMBAH URI. Masukkan domain pool pengguna Anda.

```
https://mydomain.us-east-1.amazoncognito.com
```

14. Di bawah URI pengalihan resmi, pilih TAMBAH URI. Masukkan jalur ke /oauth2/idpresponse titik akhir domain kumpulan pengguna Anda.

```
https://mydomain.us-east-1.amazoncognito.com/oauth2/idpresponse
```

15. Pilih BUAT.
16. Simpan nilai yang ditampilkan Google dengan aman di bawah ID klien Anda dan rahasia klien Anda. Berikan nilai ini ke Amazon Cognito saat Anda menambahkan Google iDP.

Untuk mendaftarkan aplikasi dengan Apple

Untuk mengetahui up-to-date informasi terbanyak tentang pengaturan Masuk dengan Apple, lihat [Mengonfigurasi Lingkungan Anda untuk Masuk dengan Apple](#) di dokumentasi Pengembang Apple.

1. Buat sebuah [akun developer dengan Apple](#).
2. [Masuk](#) dengan kredensial Apple Anda.
3. Di bilah navigasi kiri, pilih Sertifikat, Pengidentifikasi & Profil.
4. Di bilah navigasi sebelah kiri, pilih Pengidentifikasi.
5. Pada halaman Pengidentifikasi, pilih ikon +.
6. Pada halaman Daftarkan Pengidentifikasi Baru, pilih ID Aplikasi, lalu pilih Lanjutkan.

7. Pada halaman Pilih jenis, pilih Aplikasi, lalu pilih Lanjutkan.
8. Pada halaman Daftarkan ID Aplikasi, lakukan hal berikut:
  1. Di bawah Deskripsi, masukkan deskripsi.
  2. Di bawah Awalan ID Aplikasi, masukkan ID Bundel. Catat nilai di bawah Awalan ID Aplikasi. Anda akan menggunakan nilai ini setelah memilih Apple sebagai penyedia identitas Anda [Langkah 2: Tambahkan iDP sosial ke kolam pengguna Anda](#).
  3. Pada Kemampuan, pilih Masuk dengan Apple, lalu pilih Edit.
  4. Pada halaman Masuk dengan Apple: Konfigurasi ID Aplikasi, pilih untuk mengatur aplikasi sebagai primer atau dikelompokkan dengan ID Aplikasi lain, lalu pilih Simpan.
  5. Pilih Lanjutkan.
9. Pada halaman Konfirmasi ID Aplikasi Anda, pilih Daftarkan.
10. Pada halaman Pengidentifikasi, pilih ikon +.
11. Pada halaman Daftarkan Pengidentifikasi Baru, pilih ID Layanan, lalu pilih Lanjutkan.
12. Pada halaman Daftarkan ID Layanan, lakukan hal berikut:
  1. Di bawah Deskripsi, ketikkan deskripsi.
  2. Di bawah Pengidentifikasi, ketikkan pengidentifikasi. Buat catatan dari ID Layanan ini karena Anda akan memerlukan nilai ini setelah Anda memilih Apple sebagai penyedia identitas di [Langkah 2: Tambahkan iDP sosial ke kolam pengguna Anda](#).
  3. Pilih Lanjutkan, lalu Daftar.
13. Pilih ID Layanan yang baru saja Anda buat dari halaman Pengidentifikasi.
  1. Pilih Masuk dengan Apple, lalu pilih Konfigurasi.
  2. Pada halaman Konfigurasi Otentikasi Web, pilih ID aplikasi yang Anda buat sebelumnya sebagai ID Aplikasi Utama.
  3. Pilih ikon+di sebelah URL Situs Web.
  4. Di bawah Domain dan subdomain, masukkan domain kumpulan pengguna Anda tanpa awalan. `https://`

`mydomain.us-east-1.amazoncognito.com`
  5. Di bawah URL Pengembalian, masukkan jalur ke `/oauth2/idpresponse` titik akhir domain kumpulan pengguna Anda.

```
https://mydomain.us-east-1.amazoncognito.com/oauth2/idpresponse
```

6. Pilih Berikutnya, dan kemudian Selesai. Anda tidak perlu memverifikasi domain.
7. Pilih Lanjutkan, lalu pilih Simpan.
14. Di bilah navigasi sebelah kiri, pilih Kunci.
15. Pada halaman Kunci, pilih ikon +.
16. Pada halaman Daftarkan Kunci Baru, lakukan hal berikut:
  1. Di bawah Nama Kunci, masukkan nama kunci.
  2. Pilih Masuk dengan Apple, lalu pilih Konfigurasi.
  3. Pada halaman Configure Key dan pilih ID aplikasi yang Anda buat sebelumnya sebagai ID Aplikasi Utama. Pilih Simpan.
  4. Pilih Lanjutkan, lalu pilih Daftarkan.
17. Pada halaman Unduh Kunci Anda, pilih Unduh untuk mengunduh kunci pribadi dan catat ID Kunci yang ditampilkan, lalu pilih Selesai. Anda akan membutuhkan kunci privat ini dan nilai ID Kunci yang ditampilkan di halaman ini setelah Anda memilih Apple sebagai penyedia identitas di [Langkah 2: Tambahkan iDP sosial ke kolam pengguna Anda](#).

## Langkah 2: Tambahkan iDP sosial ke kolam pengguna Anda

Untuk mengkonfigurasi kumpulan pengguna sosial iDP dengan AWS Management Console

1. Masuk ke [Konsol Amazon Cognito](#). Jika diminta, masukkan AWS kredensial Anda.
2. Pilih Kolam Pengguna.
3. Pilih kolam pengguna yang ada dari daftar, atau [buat kolam pengguna](#).
4. Pilih tab Pengalaman masuk. Cari Masuk Federasi, lalu pilih Tambahkan penyedia identitas.
5. Pilih iDP sosial: Facebook, Google, Login with Amazon, atau Masuk dengan Apple.
6. Pilih dari langkah-langkah berikut, berdasarkan pilihan IDP sosial Anda:
  - Google dan Login with Amazon — Masukkan ID klien aplikasi dan rahasia klien aplikasi yang dihasilkan di bagian sebelumnya.
  - Facebook — Masukkan ID klien aplikasi dan rahasia klien aplikasi yang dihasilkan di bagian sebelumnya, lalu pilih versi API (misalnya, versi 2.12). Kami menyarankan Anda memilih versi terbaru, karena setiap API Facebook memiliki siklus hidup dan tanggal penghentian. Cakupan

dan atribut Facebook dapat bervariasi antar versi API. Kami menyarankan Anda menguji login identitas sosial Anda dengan Facebook untuk memastikan bahwa federasi berfungsi sesuai keinginan Anda.

- Masuk dengan Apple — Masukkan ID Layanan, ID Tim, ID Kunci, dan kunci pribadi yang dihasilkan di bagian sebelumnya.
7. Masukkan nama cakupan Resmi yang ingin Anda gunakan. Cakupan menentukan atribut pengguna (seperti `name` dan `email`) mana yang ingin Anda akses dengan aplikasi Anda. Untuk Facebook, ini harus dipisahkan dengan koma. Untuk Google dan Login with Amazon, mereka harus dipisahkan dengan spasi. Untuk Masuk dengan Apple, pilih kotak centang untuk cakupan yang ingin Anda akses.

Penyedia identitas sosial	Contoh cakupan
Facebook	<code>public_profile, email</code>
Google	<code>profile email openid</code>
Login with Amazon	<code>profile postal_code</code>
Masuk dengan Apple	<code>email name</code>

Pengguna aplikasi Anda diminta untuk menyetujui penyediaan atribut ini ke aplikasi Anda. Untuk informasi selengkapnya tentang cakupan penyedia sosial, lihat dokumentasi dari Google, Facebook, Login with Amazon, atau Masuk dengan Apple.

Dengan Masuk dengan Apple, berikut ini adalah skenario pengguna di mana cakupan mungkin tidak dikembalikan:

- Pengguna akhir mengalami kegagalan setelah meninggalkan halaman masuk Apple (bisa dari Kegagalan internal dalam Amazon Cognito atau apa pun yang ditulis oleh pengembang)
- ID layanan identifier digunakan di seluruh kumpulan pengguna dan/atau layanan otentikasi lainnya
- Developer menambahkan cakupan tambahan setelah pengguna akhir masuk sebelumnya (tidak ada informasi baru yang diambil)
- Developer menghapus pengguna dan kemudian pengguna masuk lagi tanpa menghapus aplikasi dari profil ID Apple mereka

8. Petakan atribut dari iDP Anda ke kumpulan pengguna Anda. Untuk informasi selengkapnya, lihat [Menentukan Pemetaan Atribut Penyedia Identitas untuk Kolam Pengguna Anda](#).
9. Pilih Buat.
10. Dari tab Integrasi klien aplikasi, pilih salah satu klien Aplikasi dalam daftar dan Edit pengaturan UI yang dihosting. Tambahkan iDP sosial baru ke klien aplikasi di bawah Penyedia identitas.
11. Pilih Simpan perubahan.

### Langkah 3: Uji konfigurasi iDP sosial Anda

Anda dapat membuat URL login dengan menggunakan elemen dari dua bagian sebelumnya. Gunakan URL login itu untuk menguji konfigurasi IdP sosial Anda.

```
https://mydomain.us-east-1.amazoncognito.com/login?  
response_type=code&client_id=1example23456789&redirect_uri=https://www.example.com
```

Anda dapat menemukan domain Anda di halaman konsol Nama Domain kolam pengguna. `client_id` ada di halaman Pengaturan klien aplikasi. Gunakan URL panggilan balik Anda untuk parameter `redirect_uri`. Ini adalah URL halaman tempat pengguna Anda akan diarahkan setelah autentikasi berhasil.

#### Note

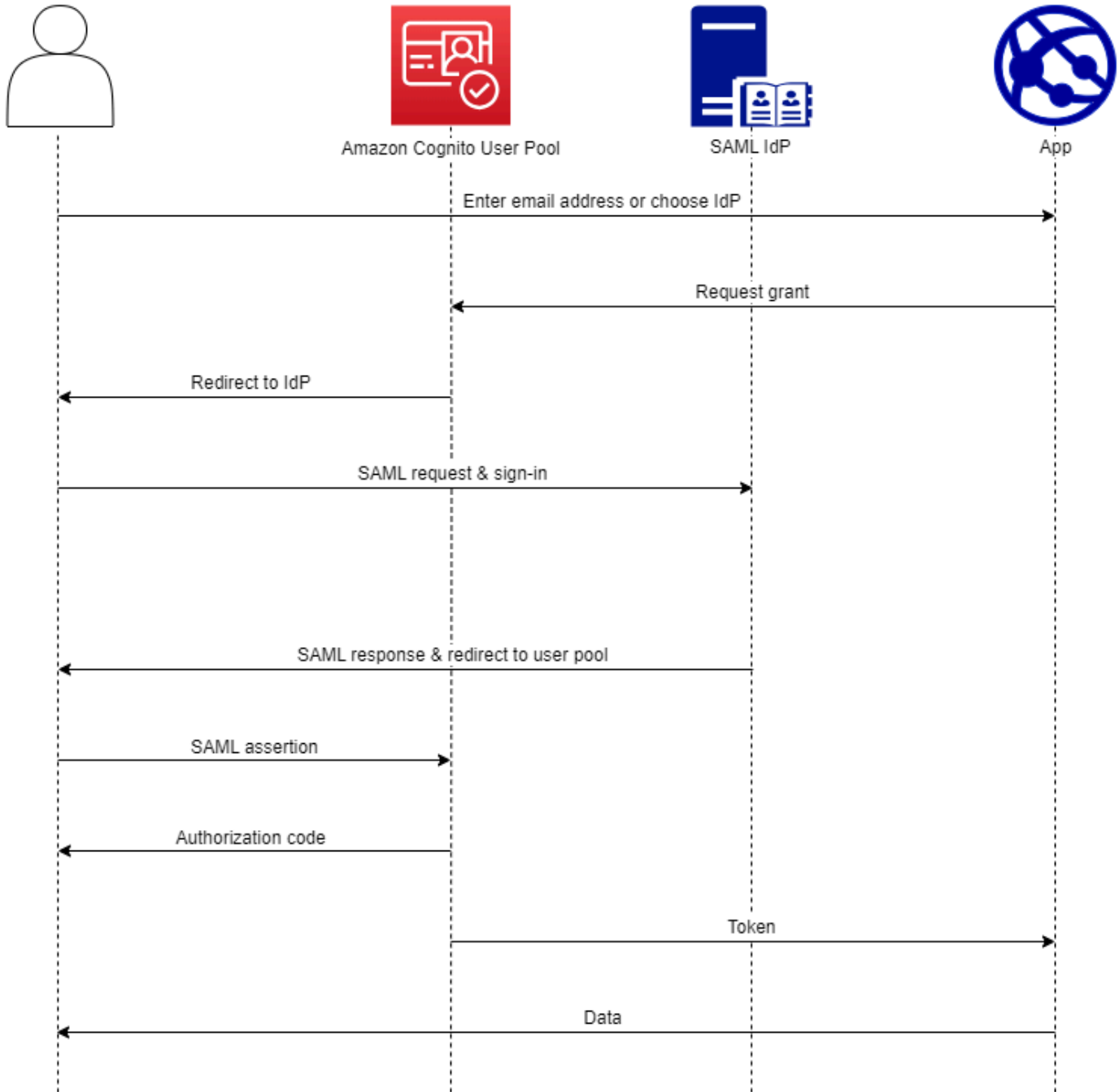
Amazon Cognito membatalkan permintaan otentikasi yang tidak selesai dalam 5 menit, dan mengarahkan pengguna ke UI yang dihosting. Halaman menampilkan pesan `Something went wrong` kesalahan.

## Menggunakan penyedia identitas SAMP dengan kumpulan pengguna

[Anda dapat memilih agar pengguna web dan aplikasi seluler masuk melalui penyedia identitas SAMP \(IDP\) seperti Microsoft Active Directory Federation Services \(ADFS\), atau Shibboleth.](#) Anda harus memilih SAMP iDP yang mendukung standar [SAMP 2.0](#).

Dengan UI yang dihosting dan titik akhir federasi, Amazon Cognito mengautentikasi pengguna IDP lokal dan pihak ketiga dan mengeluarkan token web JSON (JWT). Dengan token yang dikeluarkan Amazon Cognito, Anda dapat menggabungkan beberapa sumber identitas menjadi standar OpenID Connect (OIDC) universal di semua aplikasi Anda. Amazon Cognito dapat memproses pernyataan

SAMP dari penyedia pihak ketiga Anda ke dalam standar SSO tersebut. Anda dapat membuat dan mengelola IDP SAMP di AWS Management Console, melalui, atau dengan AWS CLI API kumpulan pengguna Amazon Cognito. Untuk membuat IDP SAMP pertama Anda di, lihat AWS Management Console. [Menambahkan dan mengelola penyedia identitas SAMP di kumpulan pengguna](#)



**Note**

Federasi dengan login melalui iDP pihak ketiga adalah fitur kumpulan pengguna Amazon Cognito. Kumpulan identitas Amazon Cognito, kadang-kadang disebut identitas federasi Amazon Cognito, adalah implementasi federasi yang harus Anda atur secara terpisah di setiap kumpulan identitas. Kumpulan pengguna dapat menjadi iDP pihak ketiga untuk kumpulan identitas. Untuk informasi selengkapnya, lihat [Kumpulan identitas Amazon Cognito](#).

## Referensi cepat untuk konfigurasi iDP

Anda harus mengkonfigurasi IDP SAMP Anda untuk menerima permintaan dan mengirim tanggapan ke kumpulan pengguna Anda. Dokumentasi untuk IDP SAMP Anda akan berisi informasi tentang cara menambahkan kumpulan pengguna Anda sebagai pihak yang mengandalkan atau aplikasi untuk IDP SAMP 2.0 Anda. Dokumentasi berikut ini memberikan nilai yang harus Anda berikan untuk URL entitas SP dan ASSERTION CONSUMER SERVICE (ACS).

Kumpulan pengguna SAMP nilai referensi cepat

ID entitas SP

```
urn:amazon:cognito:sp:us-east-1_EXAMPLE
```

URL ACS

```
https://Your user pool domain/saml2/idpresponse
```

Anda harus mengkonfigurasi kumpulan pengguna Anda untuk mendukung penyedia identitas Anda. Langkah-langkah tingkat tinggi untuk menambahkan IDP SAMP eksternal adalah sebagai berikut.

1. Unduh metadata SAMP dari IDP Anda, atau ambil URL ke titik akhir metadata Anda. Lihat [Mengonfigurasi penyedia identitas SAMP pihak ketiga Anda](#).
2. Tambahkan iDP baru ke kumpulan pengguna Anda. Unggah metadata SAMP atau berikan URL metadata. Lihat [Menambahkan dan mengelola penyedia identitas SAMP di kumpulan pengguna](#).
3. Tetapkan iDP ke klien aplikasi Anda. Lihat [Klien aplikasi kumpulan pengguna](#)

Topik



- [Hal-hal yang perlu diketahui tentang SAMP IdPs di kumpulan pengguna Amazon Cognito](#)
- [Sensitivitas kasus nama pengguna SAMP](#)
- [Menambahkan dan mengelola penyedia identitas SAMP di kumpulan pengguna](#)
- [Inisiasi sesi SAMP di kumpulan pengguna Amazon Cognito](#)
- [Menggunakan login SAMP yang diinisiasikan SP](#)
- [Menggunakan login SAMP yang diprakarsai IDP](#)
- [Alur keluar SAMP](#)
- [Penandatanganan dan enkripsi SAMP](#)
- [Nama dan pengenalan penyedia identitas SAMP](#)
- [Mengonfigurasi penyedia identitas SAMP pihak ketiga Anda](#)

## Hal-hal yang perlu diketahui tentang SAMP IdPs di kumpulan pengguna Amazon Cognito

Amazon Cognito memproses pernyataan SAMP untuk Anda

Kumpulan pengguna Amazon Cognito mendukung federasi SAMP 2.0 dengan titik akhir pasca-pengikatan. Ini menghilangkan kebutuhan aplikasi Anda untuk mengambil atau mengurai respons pernyataan SAMP, karena kumpulan pengguna langsung menerima respons SAMP dari IDP Anda melalui agen pengguna. Kolam pengguna Anda bertindak sebagai penyedia layanan (SP) atas nama aplikasi Anda. [Amazon Cognito mendukung SP-initiated dan IDP-initiated single sign-on \(SSO\) seperti yang dijelaskan di bagian 5.1.2 dan 5.1.4 dari SAMP V2.0 Technical Overview.](#)

Memberikan sertifikat penandatanganan IDP yang valid

Sertifikat penandatanganan dalam metadata penyedia SAMP Anda tidak boleh kedaluwarsa saat Anda mengonfigurasi IDP SAMP di kumpulan pengguna Anda.

Kumpulan pengguna mendukung beberapa sertifikat penandatanganan

Jika IDP SAMP Anda menyertakan lebih dari satu sertifikat penandatanganan dalam metadata SAMP, saat login, kumpulan pengguna Anda menentukan bahwa pernyataan SAMP valid jika cocok dengan sertifikat apa pun dalam metadata SAMP. Setiap sertifikat penandatanganan harus tidak lebih dari 4.096 karakter panjangnya.

Pertahankan parameter status relai

Amazon Cognito dan SAMP IDP Anda menyimpan informasi sesi dengan parameter.  
`relayState`

1. Amazon Cognito mendukung `relayState` nilai yang lebih besar dari 80 byte. Sementara spesifikasi SAMP menyatakan bahwa `relayState` nilainya “tidak boleh melebihi 80 byte panjangnya”, praktik industri saat ini sering menyimpang dari perilaku ini. Akibatnya, menolak `relayState` nilai yang lebih besar dari 80 byte akan merusak banyak integrasi penyedia SAMP standar.
2. `relayStateToken` adalah referensi buram untuk informasi negara yang dikelola oleh Amazon Cognito. Amazon Cognito tidak menjamin konten parameter. `relayState` Jangan mengurai kontennya sehingga aplikasi Anda bergantung pada hasilnya. Untuk informasi lebih lanjut, lihat [spesifikasi SAMP 2.0](#).

## Identifikasi titik akhir ACS

Penyedia identitas SAMP Anda mengharuskan Anda menetapkan titik akhir konsumen pernyataan. IDP Anda mengarahkan pengguna Anda ke titik akhir ini dengan pernyataan SAMP mereka. Konfigurasi titik akhir berikut di domain kumpulan pengguna Anda untuk pengikatan SAMP 2.0 POST di penyedia identitas SAMP Anda.

```
https://Your user pool domain/saml2/idpresponse
```

With an Amazon Cognito domain:

```
https://mydomain.us-east-1.amazoncognito.com/saml2/idpresponse
```

With a custom domain:

```
https://auth.example.com/saml2/idpresponse
```

Lihat [Mengkonfigurasi domain kolam pengguna](#) untuk informasi selengkapnya tentang domain kumpulan pengguna.

## Tidak ada pernyataan yang diputar ulang

Anda tidak dapat mengulangi, atau memutar ulang, pernyataan SAMP ke titik akhir Amazon Cognito Anda. `saml2/idpresponse` Pernyataan SAMP yang diputar ulang memiliki ID pernyataan yang menduplikasi ID dari respons IDP sebelumnya.

## ID kumpulan pengguna adalah ID entitas SP

Anda harus memberikan IDP Anda dengan ID kumpulan pengguna Anda di penyedia layanan (SP)`urn`, juga disebut URI audiens atau ID entitas SP. URI audiens untuk kumpulan pengguna Anda memiliki format berikut.

```
urn:amazon:cognito:sp:us-east-1_EXAMPLE
```

Anda dapat menemukan ID kumpulan pengguna Anda di bawah Ikhtisar kumpulan pengguna di konsol [Amazon Cognito](#).

Petakan semua atribut yang diperlukan

Konfigurasi IDP SAMP Anda untuk memberikan nilai untuk atribut apa pun yang Anda tetapkan seperti yang diperlukan dalam kumpulan pengguna Anda. Misalnya, email adalah atribut wajib umum untuk kumpulan pengguna. Sebelum pengguna dapat masuk, pernyataan IDP SAMP Anda harus menyertakan klaim bahwa Anda memetakan ke atribut kumpulan Pengguna. email Untuk informasi selengkapnya tentang pemetaan atribut, lihat [Menentukan pemetaan atribut penyedia identitas untuk kumpulan pengguna Anda](#).

Format pernyataan memiliki persyaratan khusus

IDP SAMP Anda harus menyertakan klaim berikut dalam pernyataan SAMP.

1. NameIDKlaim. Amazon Cognito mengaitkan pernyataan SAMP dengan pengguna tujuan oleh NameID. Jika NameID berubah, Amazon Cognito menganggap pernyataan tersebut untuk pengguna baru. Atribut yang Anda atur NameID dalam konfigurasi IDP Anda harus memiliki nilai persisten.

```
<saml2:NameID Format="urn:oasis:names:tc:SAML:1.1:nameid-format:persistent">
  carlos
</saml2:NameID>
```

2. AudienceRestrictionKlaim dengan Audience nilai yang menetapkan ID entitas SP kumpulan pengguna Anda sebagai target respons.

```
<saml:AudienceRestriction>
  <saml:Audience> urn:amazon:cognito:sp:us-east-1_EXAMPLE
</saml:AudienceRestriction>
```

3. Untuk sistem masuk tunggal yang dimulai SP, Response elemen dengan InResponseTo nilai ID permintaan SAMP asli.

```
<saml2p:Response Destination="https://mydomain.us-east-1.amazonaws.com/
saml2/idpresponse" ID="id123" InResponseTo="_dd0a3436-bc64-4679-
a0c2-cb4454f04184" IssueInstant="Date-time stamp" Version="2.0"
  xmlns:saml2p="urn:oasis:names:tc:SAML:2.0:protocol" xmlns:xs="http://
www.w3.org/2001/XMLSchema">
```

**Note**

Pernyataan SAMP yang diprakarsai IDP tidak boleh mengandung nilai `InResponseTo`

4. `SubjectConfirmationData` elemen dengan `Recipient` nilai `saml2/idpresponse` titik akhir kumpulan pengguna Anda dan, untuk SAMP yang diprakarsai SP, `InResponseTo` nilai yang cocok dengan ID permintaan SAMP asli.

```
<saml2:SubjectConfirmationData InResponseTo="_dd0a3436-bc64-4679-a0c2-cb4454f04184" NotOnOrAfter="Date-time stamp" Recipient="https://mydomain.us-east-1.amazoncognito.com/saml2/idpresponse"/>
```

### Permintaan masuk yang diprakarsai SP

Saat [Otorisasi titik akhir](#) mengalihkan pengguna ke halaman masuk IDP, Amazon Cognito menyertakan permintaan SAMP dalam parameter URL permintaan tersebut. HTTP GET Permintaan SAMP berisi informasi tentang kumpulan pengguna Anda, termasuk titik akhir ACS Anda. Anda dapat secara opsional menerapkan tanda tangan kriptografi untuk permintaan ini.

### Menandatangani permintaan dan mengenkripsi tanggapan

Setiap kumpulan pengguna dengan penyedia SAMP menghasilkan key pair asimetris dan sertifikat penandatanganan untuk tanda tangan digital yang ditetapkan Amazon Cognito ke permintaan SAMP. Setiap IDP SAMP eksternal yang Anda konfigurasi untuk mendukung respons SAMP terenkripsi menyebabkan Amazon Cognito menghasilkan key pair dan sertifikat enkripsi baru untuk penyedia tersebut. Untuk melihat dan mengunduh sertifikat dengan kunci publik, pilih IDP Anda di tab Pengalaman masuk di konsol Amazon Cognito.

Untuk membangun kepercayaan dengan permintaan SAMP dari kumpulan pengguna Anda, berikan IDP Anda salinan sertifikat penandatanganan SAMP 2.0 kumpulan pengguna Anda. IDP Anda mungkin mengabaikan permintaan SAMP yang ditandatangani oleh kumpulan pengguna jika Anda tidak mengonfigurasi IDP untuk menerima permintaan yang ditandatangani.

1. Amazon Cognito menerapkan tanda tangan digital untuk permintaan SAMP yang diteruskan pengguna Anda ke IDP Anda. Kumpulan pengguna Anda menandatangani semua permintaan logout tunggal (SLO), dan Anda dapat mengonfigurasi kumpulan pengguna untuk menandatangani permintaan masuk tunggal (SSO) untuk IDP eksternal SAMP apa pun. Saat Anda memberikan salinan sertifikat, IDP Anda dapat memverifikasi integritas permintaan SAMP pengguna Anda.

2. IDP SAMP Anda dapat mengenkripsi respons SAMP dengan sertifikat enkripsi. Saat Anda mengonfigurasi IDP dengan enkripsi SAMP, IDP Anda hanya harus mengirim respons terenkripsi.

### Mengkodekan karakter non-alfanumerik

Amazon Cognito tidak menerima karakter UTF-8 4-byte seperti # atau yang dilewati IDP Anda sebagai nilai atribut. Anda dapat menyandikan karakter ke Base64, meneruskannya sebagai teks, dan kemudian mendekodekannya di aplikasi Anda.

Dalam contoh berikut, klaim atribut tidak akan diterima:

```
<saml2:Attribute Name="Name" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified">
  <saml2:AttributeValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:type="xsd:string">#</saml2:AttributeValue>
</saml2:Attribute>
```

Berbeda dengan contoh sebelumnya, klaim atribut berikut akan diterima:

```
<saml2:Attribute Name="Name" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified">
  <saml2:AttributeValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:type="xsd:string">8J+YkA==</saml2:AttributeValue>
</saml2:Attribute>
```

Titik akhir metadata harus memiliki keamanan lapisan transportasi yang valid

Jika Anda melihat `InvalidParameterException` saat membuat IDP SAMP dengan URL titik akhir metadata HTTPS, misalnya, “Kesalahan mengambil metadata <metadata endpoint> dari,” pastikan bahwa titik akhir metadata memiliki SSL yang disiapkan dengan benar dan ada sertifikat SSL yang valid yang terkait dengannya. Untuk informasi selengkapnya tentang memvalidasi sertifikat, lihat [Apa itu Sertifikat SSL/TLS?](#) .

Klien aplikasi dengan SAMP yang diprakarsai IDP hanya dapat masuk dengan SAMP

Saat Anda mengaktifkan dukungan untuk IDP SAMP 2.0 yang mendukung login yang diprakarsai IDP di klien aplikasi, Anda hanya dapat menambahkan IDPs SAMP 2.0 lain ke klien aplikasi tersebut. Anda dicegah untuk menambahkan direktori pengguna di kumpulan pengguna dan semua penyedia identitas eksternal non-SAMP ke klien aplikasi yang dikonfigurasi dengan cara ini.

## Respons logout harus menggunakan pengikatan POST

/sam12/logoutTitik akhir menerima LogoutResponse sebagai HTTP POST permintaan. Kumpulan pengguna tidak menerima respons logout dengan HTTP GET pengikatan.

## Sensitivitas kasus nama pengguna SAMP

Saat pengguna federasi mencoba masuk, penyedia identitas SAMP (iDP) meneruskan unik ke Amazon NameId Cognito dalam pernyataan SAMP pengguna. Amazon Cognito mengidentifikasi pengguna yang berfederasi SAML dengan klaim mereka. NameId Terlepas dari pengaturan sensitivitas kasus kumpulan pengguna Anda, Amazon Cognito mengenali pengguna federasi yang kembali dari IDP SAMP saat mereka meneruskan klaim unik dan peka huruf besar/kecil mereka. NameId Jika Anda memetakan atribut yang email sukaNameId, dan pengguna mengubah alamat emailnya, mereka tidak dapat masuk ke aplikasi Anda.

Petakan NameId dalam pernyataan SAMP Anda dari atribut iDP yang memiliki nilai yang tidak berubah.

Misalnya, Carlos memiliki profil pengguna di kumpulan pengguna yang tidak peka huruf besar/kecil dari pernyataan SAMP Active Directory Federation Services (ADFS) yang melewati nilai. NameId Carlos@example.com Lain kali Carlos mencoba masuk, IDP ADFS Anda melewati nilai. NameId carlos@example.com Karena NameId harus sama persis dengan kasus, proses masuk tidak berhasil.

Jika pengguna Anda tidak dapat masuk setelah NameID perubahan mereka, hapus profil pengguna mereka dari kumpulan pengguna Anda. Amazon Cognito akan membuat profil pengguna baru saat mereka masuk berikutnya.

### Topik

- [Menambahkan dan mengelola penyedia identitas SAMP di kumpulan pengguna](#)
- [Inisiasi sesi SAMP di kumpulan pengguna Amazon Cognito](#)
- [Menggunakan login SAMP yang diinisiasikan SP](#)
- [Menggunakan login SAMP yang diprakarsai IDP](#)
- [Alur keluar SAMP](#)
- [Penandatanganan dan enkripsi SAMP](#)
- [Nama dan pengenalan penyedia identitas SAMP](#)
- [Mengonfigurasi penyedia identitas SAMP pihak ketiga Anda](#)

## Menambahkan dan mengelola penyedia identitas SAMP di kumpulan pengguna

Prosedur berikut menunjukkan cara membuat, memodifikasi, dan menghapus penyedia SAMP di kumpulan pengguna Amazon Cognito.

### AWS Management Console

Anda dapat menggunakan AWS Management Console untuk membuat dan menghapus penyedia identitas SAMP (IdPs).

Sebelum Anda membuat IDP SAMP, Anda harus memiliki dokumen metadata SAMP yang Anda dapatkan dari iDP pihak ketiga. Untuk petunjuk tentang cara mendapatkan atau menghasilkan dokumen metadata SAML yang diperlukan, lihat [Mengonfigurasi penyedia identitas SAMP pihak ketiga Anda](#).

Untuk mengonfigurasi SAMP 2.0 iDP di kumpulan pengguna Anda


1. Masuk ke [Konsol Amazon Cognito](#). Jika diminta, masukkan AWS kredensial Anda.
2. Pilih Kolam Pengguna.
3. Pilih kolam pengguna yang ada dari daftar, atau [buat kolam pengguna](#).
4. Pilih tab Pengalaman masuk. Cari Masuk Federasi dan pilih Tambahkan penyedia identitas.
5. Pilih SAMP iDP.
6. Masukkan nama Penyedia. Anda dapat meneruskan nama ramah ini dalam parameter `identity_provider` permintaan ke file [Otorisasi titik akhir](#).
7. Masukkan Identifier dipisahkan dengan koma. Pengidentifikasi memberi tahu Amazon Cognito bahwa ia harus memeriksa alamat email yang dimasukkan pengguna saat mereka masuk, dan kemudian mengarahkannya ke penyedia yang sesuai dengan domain mereka.
8. Pilih Tambahkan alur keluar jika Anda ingin Amazon Cognito mengirim permintaan keluar yang ditandatangani ke penyedia Anda saat pengguna keluar. Anda harus mengonfigurasi IDP SAMP 2.0 Anda untuk mengirim respons keluar ke titik akhir yang dibuat saat Anda <https://mydomain.us-east-1.amazoncognito.com/saml2/logout> mengonfigurasi UI yang dihosting. `saml2/logout`Titik akhir menggunakan pengikatan POST.

#### Note

Jika opsi ini dipilih dan IDP SALL Anda mengharapkan permintaan logout yang ditandatangani, Anda juga harus memberikan SAMP IDP Anda dengan sertifikat penandatanganan dari kumpulan pengguna Anda.

SAMP IDP akan memproses permintaan logout yang ditandatangani dan mengeluarkan pengguna Anda dari sesi Amazon Cognito.

- Pilih konfigurasi login SAMP yang diprakarsai IDP. Sebagai praktik keamanan terbaik, pilih Terima pernyataan SAMP yang diprakarsai SP saja. Jika Anda telah mempersiapkan lingkungan Anda untuk menerima sesi masuk SAMP yang tidak diminta dengan aman, pilih Terima pernyataan SAMP yang diprakarsai SP dan IDP. Untuk informasi selengkapnya, lihat [Inisiasi sesi SAMP di kumpulan pengguna Amazon Cognito](#).
- Pilih sumber dokumen Metadata. Jika IDP Anda menawarkan metadata SAMP di URL publik, Anda dapat memilih URL dokumen Metadata dan memasukkan URL publik tersebut. Jika tidak, pilih Unggah dokumen metadata dan pilih file metadata yang Anda unduh dari penyedia Anda sebelumnya.

 Note

Kami menyarankan Anda memasukkan URL dokumen metadata jika penyedia Anda memiliki titik akhir publik alih-alih mengunggah file. Amazon Cognito secara otomatis menyegarkan metadata dari URL metadata. Biasanya, penyegaran metadata terjadi setiap 6 jam atau sebelum metadata kedaluwarsa, mana yang lebih awal.

- Petakan atribut antara penyedia SAMP Anda dan kumpulan pengguna Anda untuk memetakan atribut penyedia SAMP ke profil pengguna di kumpulan pengguna Anda. Sertakan atribut wajib kumpulan pengguna Anda di peta atribut Anda.

Misalnya, ketika Anda memilih atribut `User poolemail`, masukkan nama atribut SAMP seperti yang muncul dalam pernyataan SAMP dari IDP Anda. Jika IDP Anda menawarkan contoh pernyataan SAMP, Anda dapat menggunakan pernyataan sampel ini untuk membantu Anda menemukan nama. Beberapa IDPs menggunakan nama sederhana, seperti `email`, sementara yang lain menggunakan nama seperti berikut ini.

```
http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress
```

- Pilih Buat.

## API/CLI

Gunakan perintah berikut untuk membuat dan mengelola penyedia identitas SAMP (IDP).



Untuk membuat iDP dan mengunggah dokumen metadata

- AWS CLI: `aws cognito-idp create-identity-provider`

Contoh dengan file metadata: `aws cognito-idp create-identity-provider --user-pool-id us-east-1_EXAMPLE --provider-name=SAML_provider_1 --provider-type SAML --provider-details file:///details.json --attribute-mapping email=http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress`

Di mana `details.json` berisi:

```
"ProviderDetails": {
  "MetadataFile": "<SAML metadata XML>",
  "IDPSignout" : "true",
  "RequestSigningAlgorithm" : "rsa-sha256",
  "EncryptedResponses" : "true",
  "IDPInit" : "true"
}
```

#### Note

Jika `<SAML metadata XML>` berisi instance karakter", Anda harus menambahkan `\` sebagai karakter escape: `\"`.

Contoh dengan URL metadata: `aws cognito-idp create-identity-provider --user-pool-id us-east-1_EXAMPLE --provider-name=SAML_provider_1 --provider-type SAML --provider-details MetadataURL=https://myidp.example.com/sso/saml/metadata --attribute-mapping email=http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress`

- AWS API: [CreateIdentityProvider](#)


Untuk mengunggah dokumen metadata baru untuk IDP

- AWS CLI: `aws cognito-idp update-identity-provider`

```
Contoh dengan file metadata: aws cognito-idp update-identity-provider --user-pool-id us-east-1_EXAMPLE --provider-name=SAML_provider_1 --provider-details file:///details.json --attribute-mapping email=http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress
```

Di mana details.json berisi:

```
"ProviderDetails": {
  "MetadataFile": "<SAML metadata XML>",
  "IDPSignout" : "true",
  "RequestSigningAlgorithm" : "rsa-sha256",
  "EncryptedResponses" : "true",
  "IDPInit" : "true"
}
```

 Note

Jika <SAML metadata XML> berisi instance karakter", Anda harus menambahkan \ sebagai karakter escape: \".

```
Contoh dengan URL metadata: aws cognito-idp update-identity-provider --user-pool-id us-east-1_EXAMPLE --provider-name=SAML_provider_1 --provider-details MetadataURL=https://myidp.example.com/sso/saml/metadata --attribute-mapping email=http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress
```

- AWS API: [UpdateIdentityProvider](#)

Untuk mendapatkan informasi tentang IDP tertentu

- AWS CLI: `aws cognito-idp describe-identity-provider`

```
aws cognito-idp describe-identity-provider --user-pool-id us-east-1_EXAMPLE --provider-name=SAML_provider_1
```

- AWS API: [DescribeIdentityProvider](#)

Untuk daftar informasi tentang semua IdPs

- AWS CLI: `aws cognito-idp list-identity-providers`

Contoh: `aws cognito-idp list-identity-providers --user-pool-id us-east-1_EXAMPLE --max-results 3`

- AWS API: [ListIdentityProviders](#)

Untuk menghapus IdP

- AWS CLI: `aws cognito-idp delete-identity-provider`

`aws cognito-idp delete-identity-provider --user-pool-id us-east-1_EXAMPLE --provider-name=SAML_provider_1`

- AWS API: [DeleteIdentityProvider](#)

Untuk menyiapkan IdP SAML untuk menambahkan kolam pengguna sebagai pihak yang mengandalkan

- URN penyedia layanan kolam pengguna adalah: `urn:amazon:cognito:sp:us-east-1_EXAMPLE`. Amazon Cognito memerlukan nilai pembatasan audiens yang cocok dengan URN ini dalam respons SAMP. Konfigurasi IDP Anda untuk menggunakan titik akhir pengikatan POST berikut untuk pesan respons IDP-ke-SP.

```
https://mydomain.us-east-1.amazoncognito.com/saml2/idpresponse
```

- IDP SAMP Anda harus NameID terisi dan atribut apa pun yang diperlukan untuk kumpulan pengguna Anda dalam pernyataan SAMP. NameID digunakan untuk mengidentifikasi pengguna federasi SAMP Anda secara unik di kumpulan pengguna. IDP Anda harus meneruskan ID nama SAMP setiap pengguna dalam format yang konsisten dan peka huruf besar/kecil. Setiap variasi dalam nilai ID nama pengguna membuat profil pengguna baru.

Untuk memberikan sertifikat penandatanganan ke IDP SAMP 2.0 Anda

- Untuk mengunduh salinan kunci publik dari Amazon Cognito yang dapat digunakan idP Anda untuk memvalidasi permintaan logout SAMP, pilih tab Pengalaman masuk dari kumpulan pengguna Anda, pilih IDP Anda, dan di bawah Lihat sertifikat penandatanganan, pilih Unduh sebagai.crt.

Anda dapat menghapus penyedia SAMP apa pun yang telah Anda atur di kumpulan pengguna Anda dengan konsol Amazon Cognito.

Untuk menghapus penyedia SAML

1. Masuk ke [Konsol Amazon Cognito](#).
2. Di panel navigasi, pilih Kumpulan Pengguna, dan pilih kumpulan pengguna yang ingin Anda edit.
3. Pilih tab Pengalaman masuk dan temukan login penyedia identitas Federasi.
4. Pilih tombol radio di sebelah SAMP yang ingin IdPs Anda hapus.
5. Ketika Anda diminta untuk Hapus penyedia identitas, masukkan nama penyedia SAMP untuk mengonfirmasi penghapusan, lalu pilih Hapus.

## Inisiasi sesi SAMP di kumpulan pengguna Amazon Cognito

Amazon Cognito mendukung sistem masuk tunggal (SSO) yang dimulai oleh penyedia layanan (SP-initiated) dan SSO yang diprakarsai oleh IDP. Sebagai praktik keamanan terbaik, terapkan SSO yang diprakarsai SP di kumpulan pengguna Anda. Bagian 5.1.2 dari [Ikhtisar Teknis SAMP V2.0](#) menjelaskan SSO yang diprakarsai SP. Amazon Cognito adalah penyedia identitas (iDP) ke aplikasi Anda. Aplikasi ini adalah penyedia layanan (SP) yang mengambil token untuk pengguna yang diautentikasi. Namun, ketika Anda menggunakan IDP pihak ketiga untuk mengautentikasi pengguna, Amazon Cognito adalah SP. Saat pengguna SAMP 2.0 Anda mengautentikasi dengan alur yang dimulai SP, mereka harus selalu membuat permintaan terlebih dahulu ke Amazon Cognito dan mengarahkan ulang ke IDP untuk otentikasi.

Untuk beberapa kasus penggunaan perusahaan, akses ke aplikasi internal dimulai dari bookmark di dasbor yang dihosting oleh iDP perusahaan. Ketika pengguna memilih bookmark, IDP menghasilkan respons SAMP dan mengirimkannya ke SP untuk mengautentikasi pengguna dengan aplikasi.

Anda dapat mengonfigurasi IDP SAMP di kumpulan pengguna Anda untuk mendukung SSO yang diprakarsai IDP. Saat Anda mendukung otentikasi yang diprakarsai oleh IDP, Amazon Cognito tidak dapat memverifikasi bahwa ia telah meminta respons SAMP yang diterimanya karena Amazon Cognito tidak memulai otentikasi dengan permintaan SAMP. Di SSO yang diprakarsai SP, Amazon Cognito menetapkan parameter status yang memvalidasi respons SAMP terhadap permintaan asli. Dengan login yang dimulai SP, Anda juga dapat melindungi dari pemalsuan permintaan lintas situs (CSRF).

Untuk contoh cara membangun SAMP yang diprakarsai SP di lingkungan di mana Anda tidak ingin pengguna berinteraksi dengan UI yang dihosting kumpulan pengguna, lihat. [Contoh skenario: bookmark aplikasi Amazon Cognito di dasbor perusahaan](#)

Topik

- [Contoh skenario: bookmark aplikasi Amazon Cognito di dasbor perusahaan](#)

Contoh skenario: bookmark aplikasi Amazon Cognito di dasbor perusahaan

Anda dapat membuat bookmark di dasbor SAMP atau [OIDC IDP](#) yang menyediakan akses SSO kumpulan pengguna Amazon Cognito ke aplikasi web. Anda dapat menautkan ke Amazon Cognito dengan cara yang tidak mengharuskan pengguna untuk masuk dengan UI yang dihosting. Untuk melakukannya, tambahkan bookmark masuk ke portal Anda yang mengalihkan ke kumpulan pengguna [Otorisasi titik akhir](#) Amazon Cognito Anda dalam format berikut.

```
https://mydomain.us-east-1.amazoncognito.com/authorize?  
response_type=code&identity_provider=MySAMLIdP&client_id=1example23456789&redire  
www.example.com
```

#### Note

Anda juga dapat menggunakan `idp_identifier` parameter alih-alih `identity_provider` parameter dalam permintaan Anda ke titik akhir otorisasi. IDP identifier adalah nama alternatif atau domain email yang dapat Anda konfigurasi ketika Anda membuat penyedia identitas di kumpulan pengguna Anda. Lihat [Nama dan pengenalan penyedia identitas SAMP](#).

Saat Anda menggunakan parameter yang sesuai dalam permintaan `/authorize`, Amazon Cognito secara diam-diam memulai alur masuk yang dimulai SP dan mengarahkan pengguna Anda untuk masuk dengan IDP Anda.

Untuk memulai, tambahkan IDP SAMP di kumpulan pengguna Anda. Buat klien aplikasi yang menggunakan IDP SAMP Anda untuk login, dan memiliki URL untuk aplikasi Anda sebagai URL callback resmi. Untuk informasi selengkapnya tentang klien aplikasi, lihat [Klien aplikasi kumpulan pengguna](#).

Sebelum Anda menerapkan akses terautentikasi ini ke portal Anda, uji login yang dimulai SP ke aplikasi Anda dari UI yang dihosting. Untuk informasi selengkapnya tentang cara mengonfigurasi IDP SAMP di Amazon Cognito, lihat [Mengonfigurasi penyedia identitas SAMP pihak ketiga Anda](#)

Diagram berikut menunjukkan aliran otentikasi yang mengemulasi SSO yang diprakarsai IDP. Pengguna Anda dapat mengautentikasi dengan Amazon Cognito dari tautan di portal perusahaan Anda.

Setelah Anda memenuhi persyaratan, buat bookmark untuk Anda [Otorisasi titik akhir](#) yang menyertakan parameter `identity_provider` atau `idp_identifier` parameter. Otentikasi pengguna berlangsung sebagai berikut.

1. Pengguna Anda masuk ke dasbor SSO iDP. Aplikasi perusahaan yang diizinkan oleh pengguna untuk mengakses mengisi dasbor ini.
2. Pengguna Anda memilih tautan ke aplikasi yang mengautentikasi dengan Amazon Cognito. Di banyak portal SSO, Anda dapat menambahkan tautan aplikasi khusus. Fitur apa pun yang dapat Anda gunakan untuk membuat tautan ke URL publik di portal SSO Anda akan berfungsi.
3. Tautan aplikasi kustom Anda di portal SSO mengarahkan pengguna ke kumpulan pengguna. [Otorisasi titik akhir](#) Tautan mencakup parameter `response_type`, `client_id`, `redirect_uri` dan `identity_provider`. `identity_provider` Parameternya adalah nama yang Anda berikan IDP di kumpulan pengguna Anda. Anda juga dapat menggunakan `idp_identifier` parameter alih-alih `identity_provider` parameter. Pengguna mengakses titik akhir federasi Anda dari tautan yang berisi parameter `idp_identifier` atau `identity_provider` parameter. Pengguna ini melewati halaman login dan menavigasi langsung untuk mengautentikasi dengan IDP Anda. Untuk informasi lebih lanjut tentang penamaan SAMP IdPs, lihat [Nama dan pengenalan penyedia identitas SAMP](#).

#### Contoh URL

```
https://mydomain.us-east-1.amazoncognito.com/authorize?  
response_type=code&  
identity_provider=MySAMLIdP&  
client_id=1example23456789&  
redirect_uri=https://www.example.com
```

4. Amazon Cognito mengalihkan sesi pengguna ke IDP Anda dengan permintaan SAMP.
5. Pengguna Anda mungkin telah menerima cookie sesi dari IDP Anda saat mereka masuk ke dasbor. IDP Anda menggunakan cookie ini untuk memvalidasi pengguna secara diam-diam dan mengarahkannya ke titik akhir Amazon Cognito dengan respons SAMP. `idpresponse` Jika

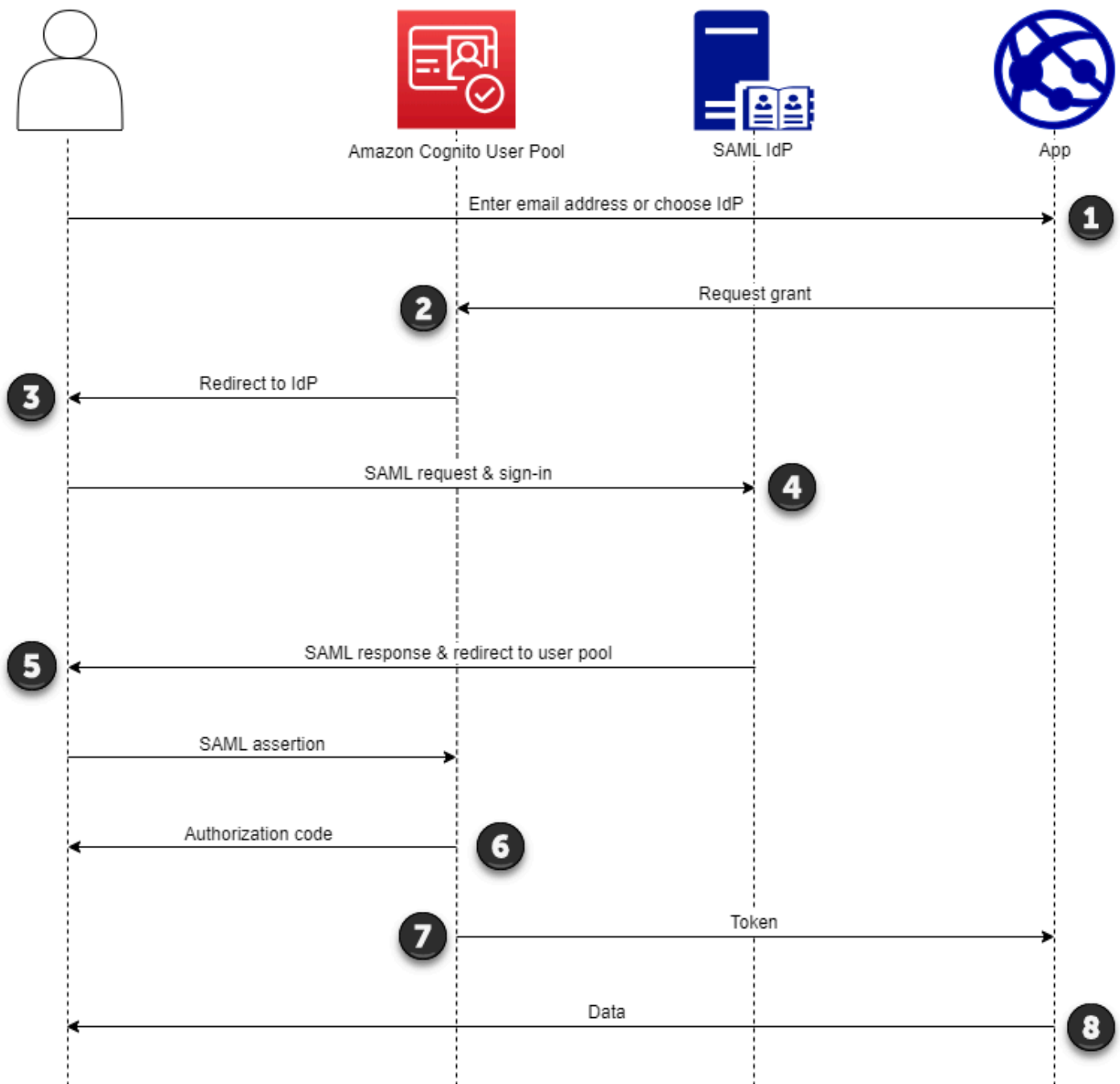
tidak ada sesi aktif, IDP Anda mengautentikasi ulang pengguna sebelum memposting respons SAMP.

6. Amazon Cognito memvalidasi respons SAMP dan membuat atau memperbarui profil pengguna berdasarkan pernyataan SAMP.
7. Amazon Cognito mengalihkan pengguna ke aplikasi internal Anda dengan kode otorisasi. Anda mengonfigurasi URL aplikasi internal sebagai URL pengalihan resmi untuk klien aplikasi Anda.
8. Aplikasi Anda menukar kode otorisasi untuk token Amazon Cognito. Untuk informasi selengkapnya, lihat [Titik akhir token](#).

## Menggunakan login SAMP yang diinisiasi SP

Sebagai praktik terbaik, terapkan login service-provider-initiated (dimulai SP) ke kumpulan pengguna Anda. Amazon Cognito memulai sesi pengguna Anda dan mengarahkannya ke IDP Anda. Dengan metode ini, Anda memiliki kendali terbesar atas siapa yang menyajikan permintaan masuk. Anda juga dapat mengizinkan login yang diprakarsai IDP dalam kondisi tertentu. Untuk informasi selengkapnya, lihat [Inisiasi sesi SAMP di kumpulan pengguna Amazon Cognito](#).

Proses berikut menunjukkan cara pengguna masuk ke kumpulan pengguna Anda melalui penyedia SAMP.



1. Pengguna Anda memasukkan alamat email mereka di halaman login. Untuk menentukan pengalihan pengguna ke IDP mereka, Anda dapat mengumpulkan alamat email mereka di aplikasi khusus atau memanggil UI yang dihosting dalam tampilan web. Anda dapat mengonfigurasi UI yang dihosting untuk menampilkan daftar IdPs atau hanya meminta alamat email.



2. Aplikasi Anda memanggil titik akhir pengalihan kumpulan pengguna dan meminta sesi dengan ID klien yang sesuai dengan aplikasi dan ID iDP yang sesuai dengan pengguna.
3. [Amazon Cognito mengalihkan pengguna Anda ke iDP dengan permintaan SAMP, yang ditandatangani secara opsional, dalam elemen AuthnRequest](#)
4. IDP mengautentikasi pengguna secara interaktif, atau dengan sesi yang diingat dalam cookie browser.
5. IDP mengarahkan pengguna Anda ke titik akhir respons SAMP kumpulan pengguna Anda dengan pernyataan SAMP yang dienkripsi secara [opsional](#) dalam muatan POST mereka.

 Note

Amazon Cognito membatalkan sesi yang tidak menerima respons dalam waktu 5 menit, dan mengarahkan pengguna ke UI yang dihosting. Ketika pengguna Anda mengalami hasil ini, mereka menerima pesan `Something went wrong` kesalahan.

6. Setelah memverifikasi pernyataan SAMP dan [memetakan atribut pengguna](#) dari klaim dalam respons, Amazon Cognito secara internal membuat atau memperbarui profil pengguna di kumpulan pengguna. Biasanya, kumpulan pengguna Anda mengembalikan kode otorisasi ke sesi browser pengguna Anda.
7. Pengguna Anda menampilkan kode otorisasi mereka ke aplikasi Anda, yang menukar kode dengan token web JSON (JWT).
8. Aplikasi Anda menerima dan memproses token ID pengguna Anda sebagai autentikasi, menghasilkan permintaan resmi ke sumber daya dengan token aksesnya, dan menyimpan token penyegaran mereka.

Saat pengguna mengautentikasi dan menerima hibah kode otorisasi, kumpulan pengguna mengembalikan ID, akses, dan token penyegaran. Token ID adalah objek otentikasi untuk manajemen identitas berbasis OIDC. Token akses adalah objek otorisasi dengan cakupan [OAuth 2.0](#). Token penyegaran adalah objek yang menghasilkan ID baru dan token akses saat token pengguna Anda saat ini telah kedaluwarsa. Anda dapat mengonfigurasi durasi token pengguna di klien aplikasi kumpulan pengguna Anda.

Anda juga dapat memilih durasi token penyegaran. Setelah token penyegaran pengguna kedaluwarsa, mereka harus masuk lagi. Jika mereka diautentikasi melalui IDP SAMP, durasi sesi pengguna Anda ditentukan oleh kedaluwarsa token mereka, bukan kedaluwarsa sesi mereka dengan IDP mereka. Aplikasi Anda harus menyimpan token penyegaran setiap pengguna dan memperbarui

sesi mereka saat kedaluwarsa. UI yang dihosting mempertahankan sesi pengguna dalam cookie browser yang berlaku selama 1 jam.

## Menggunakan login SAMP yang diprakarsai IDP

Saat mengonfigurasi penyedia identitas untuk login SAMP 2.0 yang diprakarsai IDP, Anda dapat menampilkan pernyataan SAMP ke `saml2/idpresponse` titik akhir di domain kumpulan pengguna Anda tanpa perlu memulai sesi di [Otorisasi titik akhir](#) Kumpulan pengguna dengan konfigurasi ini menerima pernyataan SAMP yang diprakarsai IDP dari penyedia identitas eksternal kumpulan pengguna yang didukung oleh klien aplikasi yang diminta. Langkah-langkah berikut menjelaskan keseluruhan proses untuk mengkonfigurasi dan masuk dengan penyedia SAMP 2.0 yang diprakarsai IDP.

1. Buat atau tentukan kumpulan pengguna dan klien aplikasi.
2. Buat SAMP 2.0 iDP di kumpulan pengguna Anda.
3. Konfigurasi IDP Anda untuk mendukung inisiasi IDP. SAMP yang diprakarsai IDP memperkenalkan pertimbangan keamanan yang tidak dikenakan oleh penyedia SSO lainnya. Karena itu, Anda tidak dapat menambahkan non-SAMP IDPs, termasuk kumpulan pengguna itu sendiri, ke klien aplikasi apa pun yang menggunakan penyedia SAMP dengan login yang diprakarsai IDP.
4. Kaitkan penyedia SAMP yang diprakarsai IDP dengan klien aplikasi di kumpulan pengguna Anda.
5. Arahkan pengguna Anda ke halaman login untuk IDP SAMP Anda dan ambil pernyataan SAMP.
6. Arahkan pengguna Anda ke `saml2/idpresponse` titik akhir kumpulan pengguna Anda dengan pernyataan SAMP mereka.
7. Terima token web JSON (JWT).

Untuk menerima pernyataan SAMP yang tidak diminta di kumpulan pengguna, Anda harus mempertimbangkan pengaruhnya terhadap keamanan aplikasi Anda. Permintaan spoofing dan upaya CSRF mungkin terjadi ketika Anda menerima permintaan yang dimulai IDP. Meskipun kumpulan pengguna Anda tidak dapat memverifikasi sesi masuk yang dimulai IDP, Amazon Cognito memvalidasi parameter permintaan dan pernyataan SAMP Anda.

Selain itu, pernyataan SAMP Anda tidak boleh mengandung `InResponseTo` klaim dan harus dikeluarkan dalam 6 menit sebelumnya.

Anda harus mengirimkan permintaan dengan SAMP yang diprakarsai IDP ke `Anda. /saml2/idpresponse` Untuk permintaan otorisasi UI yang diprakarsai SP dan yang dihosting, Anda

harus menyediakan parameter yang mengidentifikasi klien aplikasi yang diminta, cakupan, URI pengalihan, dan detail lainnya sebagai parameter string kueri dalam permintaan. HTTP GET Namun, untuk pernyataan SAMP yang diprakarsai IDP, detail permintaan Anda harus diformat sebagai RelayState parameter di badan permintaan. HTTP POST Badan permintaan juga harus berisi pernyataan SAMP Anda sebagai parameter. SAMLResponse

Berikut ini adalah contoh permintaan untuk penyedia SAMP yang diprakarsai IDP.

```
POST /saml2/idpresponse HTTP/1.1
User-Agent: USER_AGENT
Accept: */*
Host: example.auth.us-east-1.amazoncognito.com
Content-Type: application/x-www-form-urlencoded

SAMLResponse=[Base64-encoded SAML assertion]&RelayState=identity_provider
%3DMySAMLIdP%26client_id%3Dexample23456789%26redirect_uri%3Dhttps%3A%2F%2Fwww.example.com%26response_type%3Dcode%26scope%3Demail%2Bopenid%2Bphone

HTTP/1.1 302 Found
Date: Wed, 06 Dec 2023 00:15:29 GMT
Content-Length: 0
x-amz-cognito-request-id: 8aba6eb5-fb54-4bc6-9368-c3878434f0fb
Location: https://www.example.com?code=[Authorization code]
```

## AWS Management Console

Untuk mengkonfigurasi IDP untuk SAMP yang diprakarsai IDP

1. Buat [kumpulan pengguna](#), [klien aplikasi](#), dan penyedia identitas SAMP.
2. Putuskan semua penyedia identitas sosial dan OIDC dari klien aplikasi Anda, jika ada yang terkait.
3. Arahkan ke tab Pengalaman masuk di kumpulan pengguna Anda.
4. Di bawah login penyedia identitas Federasi, edit atau tambahkan penyedia SAMP.
5. Di bawah login SAMP yang diprakarsai IDP, pilih Terima pernyataan SAMP yang diprakarsai SP dan dan IDP.
6. Pilih Simpan perubahan.

## API/CLI

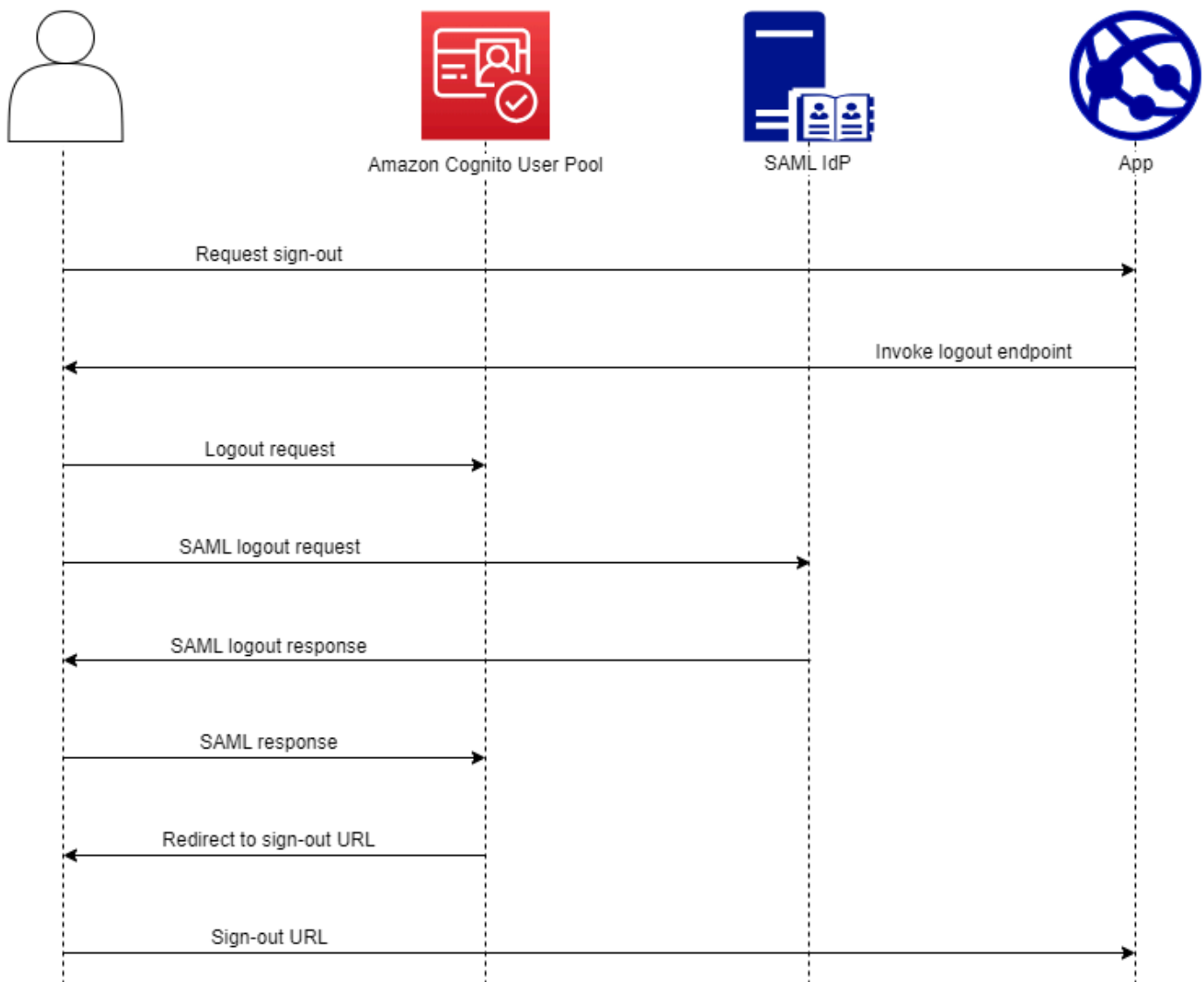
Untuk mengkonfigurasi IDP untuk SAMP yang diprakarsai IDP

Konfigurasi SAMP yang diprakarsai IDP dengan IDPInit parameter dalam permintaan atau [CreateIdentityProvider](#) API. [UpdateIdentityProvider](#) Berikut ini adalah contoh ProviderDetails dari IDP yang mendukung IDP yang diprakarsai SAMP.

```
"ProviderDetails": {
  "MetadataURL" : "https://myidp.example.com/saml/metadata",
  "IDPSignout" : "true",
  "RequestSigningAlgorithm" : "rsa-sha256",
  "EncryptedResponses" : "true",
  "IDPInit" : "true"
}
```

## Alur keluar SAMP

[Amazon Cognito mendukung logout tunggal SAMP 2.0](#). Saat Anda mengonfigurasi IDP SAMP untuk mendukung alur Keluar, Amazon Cognito mengalihkan pengguna Anda dengan permintaan logout SAMP yang ditandatangani ke iDP Anda. Amazon Cognito menentukan lokasi pengalihan dari SingleLogoutService URL di metadata iDP Anda. Amazon Cognito menandatangani permintaan keluar dengan sertifikat penandatanganan kumpulan pengguna Anda.



Saat Anda mengarahkan pengguna dengan sesi SAMP ke `/logout` titik akhir kumpulan pengguna, Amazon Cognito mengalihkan pengguna SAMP Anda dengan permintaan berikut ke titik akhir SLO yang ditentukan dalam metadata iDP.

```

https://[SingleLogoutService endpoint]?
SAMLRequest=[encoded SAML request]&
RelayState=[RelayState]&
SigAlg=http://www.w3.org/2001/04/xmldsig-more#rsa-sha256&
Signature=[User pool RSA signature]
  
```

Pengguna Anda kemudian kembali ke `saml2/logout` titik akhir Anda dengan a LogoutResponse dari IDP mereka. IDP Anda harus mengirimkan LogoutResponse permintaan. HTTP POST Amazon Cognito kemudian mengarahkan mereka ke tujuan pengalihan dari permintaan keluar awal mereka.

Penyedia SAFL Anda mungkin mengirim LogoutResponse dengan lebih dari satu AuthnStatement di dalamnya. Yang pertama `sessionIndex` AuthnStatement dalam respons jenis ini harus cocok dengan respons SAMP `sessionIndex` yang awalnya mengautentikasi pengguna. Jika `sessionIndex` ada di tempat lainAuthnStatement, Amazon Cognito tidak akan mengenali sesi tersebut dan pengguna Anda tidak akan keluar.

## AWS Management Console

Untuk mengonfigurasi keluar SAMP

1. Buat [kumpulan pengguna](#), [klien aplikasi](#), dan SAMP iDP.
2. Saat Anda membuat atau mengedit penyedia identitas SAMP Anda, di bawah Informasi penyedia identitas, centang kotak dengan judul Tambahkan alur keluar.
3. Dari tab Pengalaman masuk di kumpulan pengguna Anda, di bawah login penyedia identitas Federasi, pilih IDP Anda dan temukan sertifikat Penandatanganan.
4. Pilih Unduh sebagai.crt.
5. Konfigurasi penyedia SAMP Anda untuk mendukung logout tunggal SAMP dan penandatanganan permintaan, dan unggah sertifikat penandatanganan kumpulan pengguna. IDP Anda harus mengarahkan ulang ke domain kumpulan `/saml2/logout` pengguna Anda.

## API/CLI

Untuk mengonfigurasi keluar SAMP

Konfigurasi logout tunggal dengan IDPSignout parameter permintaan [CreateIdentityProvider](#) atau [UpdateIdentityProvider](#) API. Berikut ini adalah contoh `ProviderDetails` dari iDP yang mendukung SAMP single logout.

```
"ProviderDetails": {
  "MetadataURL" : "https://myidp.example.com/saml/metadata",
  "IDPSignout" : "true",
  "RequestSigningAlgorithm" : "rsa-sha256",
  "EncryptedResponses" : "true",
  "IDPInit" : "true"
```

```
}
```

## Penandatanganan dan enkripsi SAMP

Amazon Cognito mendukung permintaan SAMP yang ditandatangani dan respons SAMP terenkripsi untuk masuk dan keluar. Semua operasi kriptografi selama operasi SAMP kumpulan pengguna harus menghasilkan tanda tangan dan ciphertext dengan kunci yang user-pool-provided dihasilkan Amazon Cognito. Saat ini, Anda tidak dapat mengonfigurasi kumpulan pengguna untuk menandatangani permintaan atau menerima pernyataan terenkripsi dengan kunci eksternal.

### Note

Sertifikat kumpulan pengguna Anda berlaku selama 10 tahun. Sekali setahun, Amazon Cognito menghasilkan sertifikat penandatanganan dan enkripsi baru untuk kumpulan pengguna Anda. Amazon Cognito mengembalikan sertifikat terbaru saat Anda meminta sertifikat penandatanganan, dan menandatangani permintaan dengan sertifikat penandatanganan terbaru. IDP Anda dapat mengenkripsi pernyataan SAMP dengan sertifikat enkripsi kumpulan pengguna apa pun yang tidak kedaluwarsa. Sertifikat Anda sebelumnya tetap berlaku untuk seluruh durasinya. Sebagai praktik terbaik, perbarui sertifikat dalam konfigurasi penyedia Anda setiap tahun.

### Topik

- [Menerima tanggapan SAMP terenkripsi dari IDP Anda](#)
- [Menandatangani permintaan SAMP](#)

### Menerima tanggapan SAMP terenkripsi dari IDP Anda

Amazon Cognito dan IDP Anda dapat menetapkan kerahasiaan dalam respons SAMP saat pengguna masuk dan keluar. Amazon Cognito menetapkan key pair RSA publik-pribadi dan sertifikat ke setiap penyedia SAMP eksternal yang Anda konfigurasi di kumpulan pengguna. Saat mengaktifkan enkripsi respons untuk penyedia SAMP kumpulan pengguna, Anda harus mengunggah sertifikat ke IDP yang mendukung respons SAMP terenkripsi. Koneksi kumpulan pengguna Anda ke SAMP IDP Anda tidak berfungsi sebelum IDP Anda mulai mengenkripsi semua pernyataan SAMP dengan kunci yang disediakan.

Berikut ini adalah ikhtisar aliran masuk SAMP terenkripsi.

1. Pengguna Anda mulai masuk dan memilih SAMP iDP mereka.
2. Kumpulan pengguna Anda [Otorisasi titik akhir](#) mengarahkan pengguna Anda ke IDP SAMP mereka dengan permintaan masuk SAMP. Kumpulan pengguna Anda dapat secara opsional menyertai permintaan ini dengan tanda tangan yang memungkinkan verifikasi integritas oleh iDP. Ketika Anda ingin menandatangani permintaan SAMP, Anda harus mengonfigurasi IDP Anda untuk menerima permintaan yang telah ditandatangani oleh kumpulan pengguna Anda dengan kunci publik dalam sertifikat penandatanganan.
3. IDP SAMP masuk ke pengguna Anda dan menghasilkan respons SAMP. IDP mengenkripsi respons dengan kunci publik dan mengarahkan pengguna Anda ke titik akhir kumpulan pengguna Anda. `/sam12/idpresponse` IDP harus mengenkripsi respons seperti yang didefinisikan oleh spesifikasi SAMP 2.0. Untuk informasi lebih lanjut, lihat Element `<EncryptedAssertion>` di Pernyataan [dan Protokol untuk OASIS Security Assertion Markup Language \(SAMP\) V2.0](#).
4. Kumpulan pengguna Anda mendekripsi ciphertext dalam respons SAMP dengan kunci pribadi dan tanda di pengguna Anda.

#### Important

Saat Anda mengaktifkan enkripsi respons untuk IDP SAMP di kumpulan pengguna, iDP Anda harus mengenkripsi semua respons dengan kunci publik yang khusus untuk penyedia. Amazon Cognito tidak menerima respons SAMP yang tidak terenkripsi dari IDP eksternal SAMP yang Anda konfigurasi untuk mendukung enkripsi.

Setiap IDP SAMP eksternal di kumpulan pengguna Anda dapat mendukung enkripsi respons, dan setiap iDP menerima key pair sendiri.

## AWS Management Console

Untuk mengkonfigurasi enkripsi respons SAMP

1. Buat [kumpulan pengguna](#), [klien aplikasi](#), dan SAMP iDP.
2. Saat Anda membuat atau mengedit penyedia identitas SAMP Anda, di bawah Menandatangani permintaan dan mengenkripsi tanggapan, centang kotak dengan judul Memerlukan pernyataan SAMP terenkripsi dari penyedia ini.
3. Dari tab Pengalaman masuk di kumpulan pengguna Anda, di bawah login penyedia identitas Federasi, pilih IDP SAMP Anda dan pilih Lihat sertifikat enkripsi.



4. Pilih Unduh sebagai.crt dan berikan file yang diunduh ke SAMP IDP Anda. Konfigurasi IDP SAMP Anda untuk mengenkripsi respons SAMP dengan kunci dalam sertifikat.

## API/CLI

Untuk mengkonfigurasi enkripsi respons SAMP

Konfigurasi enkripsi respons dengan EncryptedResponses parameter permintaan [CreateIdentityProvider](#) atau [UpdateIdentityProvider](#) API. Berikut ini adalah contoh IDP ProviderDetails yang mendukung penandatanganan permintaan.

```
"ProviderDetails": {
  "MetadataURL" : "https://myidp.example.com/saml/metadata",
  "IDPSignout" : "true",
  "RequestSigningAlgorithm" : "rsa-sha256",
  "EncryptedResponses" : "true",
  "IDPInit" : "true"
}
```

## Menandatangani permintaan SAMP

Kemampuan untuk membuktikan integritas permintaan SAMP 2.0 ke IDP Anda adalah keuntungan keamanan dari login SAMP yang diprakarsai Amazon Cognito SP. Setiap kumpulan pengguna dengan domain menerima sertifikat penandatanganan kumpulan pengguna X.509. Dengan kunci publik dalam sertifikat ini, kumpulan pengguna menerapkan tanda tangan kriptografi ke permintaan keluar yang dihasilkan oleh kumpulan pengguna Anda saat pengguna memilih IDP SAMP. Anda dapat mengonfigurasi klien aplikasi secara opsional untuk menandatangani permintaan masuk SAMP. Saat Anda menandatangani permintaan SAMP, IDP Anda dapat memeriksa apakah tanda tangan dalam metadata XHTML permintaan Anda cocok dengan kunci publik dalam sertifikat kumpulan pengguna yang Anda berikan.

## AWS Management Console

Untuk mengonfigurasi penandatanganan permintaan SAMP

1. Buat [kumpulan pengguna](#), [klien aplikasi](#), dan SAMP iDP.
2. Saat Anda membuat atau mengedit penyedia identitas SAMP Anda, di bawah Menandatangani permintaan dan mengenkripsi tanggapan, centang kotak dengan judul Tanda tangani permintaan SAMP ke penyedia ini.

3. Dari tab Pengalaman masuk kumpulan pengguna Anda, di bawah login penyedia identitas Federasi, pilih Lihat sertifikat penandatanganan.
4. Pilih Unduh sebagai.crt dan berikan file yang diunduh ke SAMP IDP Anda. Konfigurasi IDP SAMP Anda untuk memverifikasi tanda tangan permintaan SAMP yang masuk.

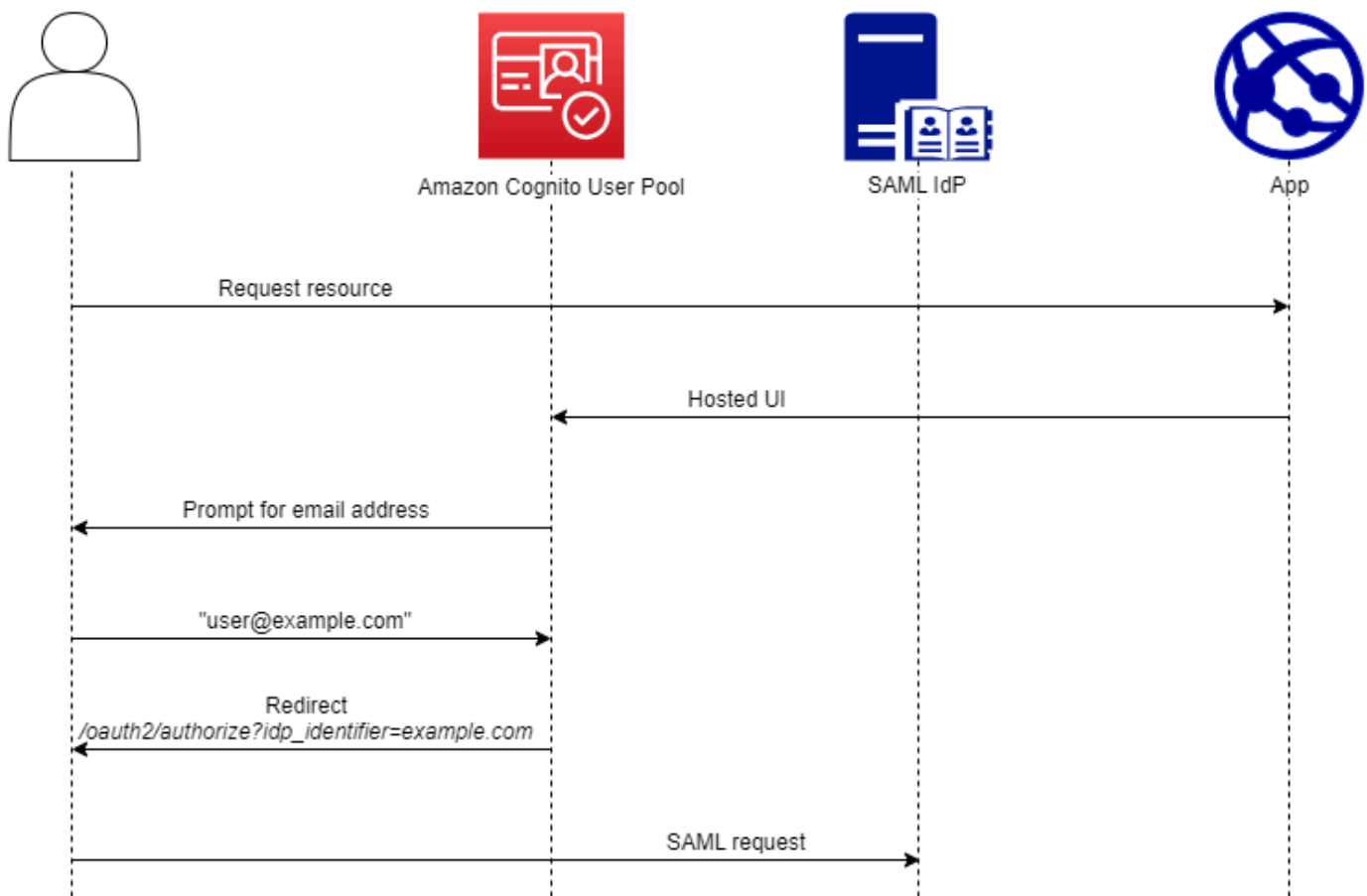
## API/CLI

Untuk mengonfigurasi penandatanganan permintaan SAMP

Konfigurasi penandatanganan permintaan dengan `RequestSigningAlgorithm` parameter permintaan [CreateIdentityProvider](#) atau [UpdateIdentityProvider](#) API. Berikut ini adalah contoh IDP `ProviderDetails` yang mendukung penandatanganan permintaan.

```
"ProviderDetails": {
  "MetadataURL" : "https://myidp.example.com/saml/metadata",
  "IDPSignout" : "true",
  "RequestSigningAlgorithm" : "rsa-sha256",
  "EncryptedResponses" : "true",
  "IDPInit" : "true"
}
```

## Nama dan pengenalan penyedia identitas SAMP



Saat Anda memberi nama penyedia identitas SAMP (IdPs) dan menetapkan pengidentifikasi IDP, Anda dapat mengotomatiskan alur permintaan masuk dan keluar yang dimulai SP ke penyedia tersebut. Untuk informasi tentang batasan string ke nama penyedia, lihat properti. `ProviderName` [CreateIdentityProvider](#)

Anda juga dapat memilih hingga 50 pengidentifikasi untuk penyedia SAMP Anda. Pengenal adalah nama yang ramah untuk IDP di kumpulan pengguna Anda, dan harus unik di dalam kumpulan pengguna. Jika pengidentifikasi SAMP Anda cocok dengan domain email pengguna Anda, UI yang dihosting Amazon Cognito akan meminta alamat email setiap pengguna, mengevaluasi domain di alamat email mereka, dan mengarahkannya ke IDP yang sesuai dengan domain mereka. Karena organisasi yang sama dapat memiliki beberapa domain, satu IDP dapat memiliki beberapa pengidentifikasi.

Baik Anda menggunakan atau tidak menggunakan pengenal email-domain, Anda dapat menggunakan pengenal di aplikasi multi-penyewa untuk mengarahkan pengguna ke IDP yang benar.

Saat Anda ingin melewati UI yang dihosting sepenuhnya, Anda dapat menyesuaikan tautan yang Anda sajikan kepada pengguna sehingga mereka mengarahkan [Otorisasi titik akhir](#) langsung ke IDP mereka. Untuk masuk ke pengguna Anda dengan pengenalan dan mengarahkan ulang ke IDP mereka, sertakan pengenalan dalam format `idp_identifier=myidp.example.com` dalam parameter permintaan otorisasi awal mereka.

Metode lain untuk meneruskan pengguna ke IDP Anda adalah untuk mengisi parameter `identity_provider` dengan nama IDP Anda dalam format URL berikut.

```
https://mydomain.us-east-1.amazoncognito.com/oauth2/authorize?
response_type=code&
identity_provider=MySAMLIdP&
client_id=1example23456789&
redirect_uri=https://www.example.com
```

Setelah pengguna masuk dengan SAMP IDP Anda, IDP Anda mengarahkan mereka dengan respons SAMP di badan ke titik akhir Anda. HTTP POST `/saml2/idpresponse` Amazon Cognito memproses pernyataan SAMP dan, jika klaim dalam respons memenuhi harapan, mengalihkan ke URL panggilan balik klien aplikasi Anda. Setelah pengguna Anda menyelesaikan autentikasi dengan cara ini, mereka berinteraksi dengan halaman web hanya untuk IDP dan aplikasi Anda.

Dengan pengidentifikasi IDP dalam format domain, UI yang di-host Amazon Cognito meminta alamat email saat login dan kemudian, ketika domain email cocok dengan pengenalan IDP, mengarahkan pengguna ke halaman login untuk IDP mereka. Misalnya, Anda membuat aplikasi yang memerlukan login oleh karyawan dari dua perusahaan yang berbeda. Perusahaan pertama, AnyCompany A, memiliki `exampleA.com` dan `exampleA.co.uk`. Perusahaan kedua, AnyCompany B, memiliki `exampleB.com`. Untuk contoh ini, Anda telah menyiapkan dua IDPs, satu untuk setiap perusahaan, sebagai berikut:

- Untuk IDP A, Anda menentukan pengidentifikasi `exampleA.com` dan `exampleA.co.uk`
- Untuk IDP B, Anda mendefinisikan identifier `exampleB.com`

Di aplikasi Anda, panggil UI yang dihosting untuk klien aplikasi Anda untuk meminta setiap pengguna memasukkan alamat email mereka. Amazon Cognito memperoleh domain dari alamat email, menghubungkan domain ke IDP dengan pengenalan domain, dan mengarahkan pengguna Anda ke IDP yang benar dengan permintaan ke yang berisi parameter permintaan.

[Otorisasi titik akhir](#) `idp_identifier` Misalnya, jika pengguna masuk `bob@exampleA.co.uk`,

halaman berikutnya yang berinteraksi dengan mereka adalah halaman masuk IDP di. <https://auth.exampleA.co.uk/sso/saml>

Anda juga dapat menerapkan logika yang sama secara independen. Di aplikasi Anda, Anda dapat membuat formulir kustom yang mengumpulkan input pengguna dan menghubungkannya dengan iDP yang benar sesuai dengan logika Anda sendiri. Anda dapat membuat portal aplikasi khusus untuk setiap penyewa aplikasi Anda, di mana setiap link ke titik akhir otorisasi dengan pengenalan penyewa dalam parameter permintaan.

Untuk mengumpulkan alamat email dan mengurai domain di UI yang dihosting, tetapkan setidaknya satu pengenalan ke setiap IDP SAMP yang telah ditetapkan ke klien aplikasi Anda. Secara default, layar login UI yang dihosting menampilkan tombol untuk setiap tombol IdPs yang telah ditetapkan ke klien aplikasi. Namun, jika Anda berhasil menetapkan pengenalan, halaman login UI yang dihosting akan terlihat seperti gambar berikut.

Penguraian domain di UI yang dihosting mengharuskan Anda menggunakan domain sebagai pengidentifikasi IDP Anda. Jika Anda menetapkan pengenalan jenis apa pun ke masing-masing SAMP untuk klien aplikasi, UI yang dihosting IdPs untuk aplikasi tersebut tidak lagi menampilkan tombol pemilihan IDP. Tambahkan pengidentifikasi IDP untuk SAMP saat Anda bermaksud menggunakan penguraian email atau logika khusus untuk menghasilkan pengalihan. Saat Anda ingin membuat pengalihan senyap dan juga ingin UI yang dihosting menampilkan daftar IdPs, jangan tetapkan pengenalan dan gunakan parameter `identity_provider` permintaan dalam permintaan otorisasi Anda.

- Jika Anda menetapkan hanya satu IDP SAMP ke klien aplikasi Anda, halaman login UI yang dihosting akan menampilkan tombol untuk masuk dengan iDP tersebut.
- Jika Anda menetapkan pengenalan ke setiap IDP SAMP yang Anda aktifkan untuk klien aplikasi, prompt input pengguna untuk alamat email akan muncul di halaman login UI yang dihosting.
- Jika Anda memiliki beberapa IdPs dan Anda tidak menetapkan pengenalan untuk semuanya, halaman login UI yang dihosting akan menampilkan tombol untuk masuk dengan setiap IDP yang ditetapkan.
- Jika Anda menetapkan pengenalan ke Anda IdPs dan Anda ingin UI yang dihosting menampilkan pilihan tombol iDP, tambahkan iDP baru yang tidak memiliki pengenalan ke klien aplikasi Anda, atau buat klien aplikasi baru. Anda juga dapat menghapus IDP yang ada dan menambahkannya lagi tanpa pengenalan. Jika Anda membuat iDP baru, pengguna SAMP Anda akan membuat profil pengguna baru. Duplikasi pengguna aktif ini mungkin memiliki dampak penagihan di bulan saat Anda mengubah konfigurasi iDP.

Untuk informasi lebih lanjut tentang persiapan IdP, lihat [Mengkonfigurasi penyedia identitas untuk kumpulan pengguna Anda](#).

## Mengonfigurasi penyedia identitas SAMP pihak ketiga Anda

Untuk mengonfigurasi solusi penyedia identitas SAMP 2.0 (iDP) pihak ketiga agar berfungsi dengan federasi untuk kumpulan pengguna Amazon Cognito, Anda harus mengonfigurasi IDP SAMP untuk mengarahkan ulang ke URL Assertion Consumer Service (ACS) berikut: `https://mydomain.us-east-1.amazoncognito.com/saml2/idpresponse` Jika kumpulan pengguna memiliki domain Amazon Cognito, Anda dapat menemukan jalur domain kumpulan pengguna di tab Integrasi aplikasi kumpulan pengguna di konsol [Amazon Cognito](#).

Beberapa SAMP IdPs mengharuskan Anda memberikan urn, juga disebut URI audiens atau ID entitas SP, dalam formulirurn:amazon:cognito:sp:`us-east-1_EXAMPLE`. Anda dapat menemukan ID kumpulan pengguna Anda di bawah Ikhtisar kumpulan pengguna di konsol Amazon Cognito.

Anda juga harus mengonfigurasi SAMP IDP Anda untuk memberikan nilai untuk atribut apa pun yang Anda tetapkan sebagai atribut wajib di kumpulan pengguna Anda. Biasanya, email adalah atribut wajib untuk kumpulan pengguna, dalam hal ini IDP SAMP harus memberikan beberapa bentuk email klaim dalam pernyataan SAMP mereka, dan Anda harus memetakan klaim ke atribut untuk penyedia tersebut.

Informasi konfigurasi berikut untuk solusi IDP SAMP 2.0 pihak ketiga adalah tempat yang baik untuk mulai menyiapkan federasi dengan kumpulan pengguna Amazon Cognito. Untuk informasi terbaru, konsultasikan dokumentasi penyedia Anda secara langsung.

Untuk menandatangani permintaan SAMP, Anda harus mengonfigurasi idP Anda untuk mempercayai permintaan yang ditandatangani oleh sertifikat penandatanganan kumpulan pengguna Anda. Untuk menerima respons SAMP terenkripsi, Anda harus mengonfigurasi IDP Anda untuk mengenkripsi semua respons SAMP ke kumpulan pengguna Anda. Penyedia Anda akan memiliki dokumentasi tentang mengonfigurasi fitur-fitur ini. Untuk contoh dari Microsoft, lihat [Mengkonfigurasi enkripsi token Microsoft Entra SAMP](#).

### Note

Amazon Cognito hanya memerlukan dokumen metadata penyedia identitas Anda. Penyedia Anda mungkin menawarkan informasi konfigurasi untuk Akun AWS federasi dengan SAMP 2.0; informasi ini tidak relevan dengan integrasi Amazon Cognito.

Solusi	Informasi selengkapnya
Microsoft Active Directory Federation Services (AD FS)	<a href="#">Federasi Metadata Explorer</a>
Okta	<a href="#">Cara Mengunduh Metadata iDP dan Sertifikat Penandatanganan SAMP untuk Integrasi Aplikasi SAMP</a>
Auth0	<a href="#">Konfigurasi Auth0 sebagai Penyedia Identitas SAMP</a>
Identitas Ping (PingFederate)	<a href="#">Mengekspor metadata SAMP dari PingFederate</a>
JumpCloud	<a href="#">Catatan Konfigurasi SAMP</a>
SecureAuth	<a href="#">Integrasi aplikasi SAMP</a>

## Menggunakan penyedia identitas OIDC dengan kumpulan pengguna

Anda dapat mengaktifkan pengguna yang sudah memiliki akun dengan penyedia identitas [OpenID Connect \(OIDC\)](#) (IdPs) untuk melewati langkah pendaftaran dan masuk ke aplikasi menggunakan akun yang sudah ada. Dengan UI web host bawaan, Amazon Cognito menyediakan penanganan dan manajemen token untuk semua pengguna yang diautentikasi. Dengan cara ini, sistem backend Anda dapat melakukan standarisasi pada satu set token kumpulan pengguna.



### Note

Masuk melalui pihak ketiga (federasi) tersedia di kolam pengguna Amazon Cognito. Fitur ini independen dari federasi melalui kolam identitas Amazon Cognito (identitas federasi).

Anda dapat menambahkan IdP OIDC ke kumpulan pengguna Anda di AWS Management Console, melalui AWS CLI, atau dengan metode API kumpulan pengguna. [CreateIdentityProvider](#)

## Topik

- [Prasyarat](#)
- [Langkah 1: Mendaftar dengan IdP OIDC](#)
- [Langkah 2: Tambahkan OIDC iDP ke kumpulan pengguna Anda](#)
- [Langkah 3: Uji konfigurasi IDP OIDC Anda](#)
- [Aliran otentikasi IDP kumpulan pengguna OIDC](#)

## Prasyarat

Sebelum memulai, Anda perlu melakukan hal berikut:

- Kumpulan pengguna dengan klien aplikasi dan domain kumpulan pengguna. Untuk informasi selengkapnya, lihat [Membuat kolam pengguna](#).
- IdP OIDC dengan konfigurasi berikut:
  - Mendukung otentikasi `client_secret_post` klien. Amazon Cognito tidak memeriksa `token_endpoint_auth_methods_supported` klaim di titik akhir penemuan OIDC untuk IDP Anda. Amazon Cognito tidak mendukung otentikasi `client_secret_basic` klien. Untuk informasi selengkapnya tentang otentikasi klien, lihat [Otentikasi Klien di dokumentasi](#) OpenID Connect.
  - Hanya menggunakan HTTPS untuk endpoint OIDC seperti `openid_configuration`, `userInfo` dan `jwt_uri`
  - Hanya menggunakan port TCP 80 dan 443 untuk titik akhir OIDC.
  - Hanya menandatangani token ID dengan algoritma HMAC-SHA, ECDSA, atau RSA.
  - Menerbitkan `kid` klaim ID kunci di dalamnya `jwt_uri` dan menyertakan `kid` klaim dalam tokennya.

## Langkah 1: Mendaftar dengan IdP OIDC

Sebelum Anda membuat IdP OIDC dengan Amazon Cognito, Anda harus mendaftarkan aplikasi Anda dengan IdP OIDC untuk menerima ID klien dan rahasia klien.




## Untuk mendaftar dengan IdP OIDC

1. Buat akun developer dengan IdP OIDC.

### Tautan ke OIDC IdPs

IdP OIDC	Cara Menginstal	URL Penemuan OIDC
Salesforce	<a href="#">Instal penyedia identitas Salesforce</a>	https://login.salesforce.com
Identitas Ping	<a href="#">Instal penyedia identitas Ping Identity</a>	https:// <i>Alamat domain Ping Anda</i> :9031/idp/userinfo.openid  Misalnya: https://p f.company.com:9031/ idp/userinfo.openid
Okta	<a href="#">Instal penyedia identitas Okta</a>	https:// <i>Subdomain Okta Anda</i> .oktapreview.com  atau https:// <i>Your Okta subdomain</i> .okta.com
Microsoft Azure Active Directory (Azure AD)	<a href="#">Instal penyedia identitas Microsoft Azure AD</a>	https://login.microsoftonline.com/ <i>{tenant}</i> /v2.0

IdP OIDC	Cara Menginstal	URL Penemuan OIDC
Google	<a href="#">Instal penyedia identitas Google</a>	https://accounts.google.com

 **Note**

Amazon Cognito menawarkan Google sebagai IdP masuk sosial yang terintegrasi. Sebaiknya Anda menggunakan IdP terintegrasi. Lihat [Menggunakan penyedia identitas sosial dengan kumpulan pengguna](#).

- Daftarkan URL domain pool pengguna Anda dengan /oauth2/idpresponse endpoint dengan IDP OIDC Anda. Hal ini memastikan bahwa IdP OIDC nanti menerimanya dari Amazon Cognito ketika mengautentikasi pengguna.

`https://mydomain.us-east-1.amazoncognito.com/oauth2/idpresponse`

- Daftarkan URL panggilan balik Anda dengan kumpulan pengguna Amazon Cognito Anda. Ini adalah URL halaman tempat Amazon Cognito mengalihkan pengguna Anda setelah otentikasi berhasil.

`https://www.example.com`

- Pilih [cakupan](#) Anda. Cakupan openid diperlukan. Cakupan email diperlukan untuk memberikan akses ke [klaim](#) email dan email\_verified.
- IdP OIDC memberi Anda ID klien dan rahasia klien. Anda akan menggunakannya saat menyiapkan IdP OIDC di kolam pengguna Anda.

Contoh: Gunakan Salesforce sebagai IdP OIDC dengan kolam pengguna Anda

Anda menggunakan IdP OIDC ketika Anda ingin membangun kepercayaan antara IDP yang kompatibel dengan OIDC seperti Salesforce dan kumpulan pengguna Anda.

1. [Buat akun](#) di situs web Developer Salesforce.
2. [Masuk](#) melalui akun developer yang Anda siapkan di langkah sebelumnya.
3. Dari halaman Salesforce Anda, lakukan salah satu hal berikut:
  - Jika Anda menggunakan Lightning Experience, pilih ikon setup gear, lalu pilih Setup Home.
  - Jika Anda menggunakan Salesforce Classic dan Anda melihat Penyiapan di header antarmuka pengguna, pilih itu.
  - Jika Anda menggunakan Salesforce Classic dan Anda tidak melihat Penyiapan di header, pilih nama Anda dari bilah navigasi atas, lalu pilih Penyiapan dari daftar tarik-turun.
4. Di bilah navigasi kiri, pilih Pengaturan Perusahaan.
5. Pada bilah navigasi, pilih Domain, masukkan domain, dan pilih Buat.
6. Di bilah navigasi kiri, di bawah Alat Platform, pilih Aplikasi.
7. Pilih Pengelola Aplikasi.
8.
  - a. Pilih Aplikasi baru yang terhubung.
  - b. Lengkapi bidang yang diperlukan.

Di bawah URL Mulai, masukkan URL di `/authorize` titik akhir untuk domain kumpulan pengguna yang masuk dengan IDP Salesforce Anda. Saat pengguna mengakses aplikasi yang terhubung, Salesforce mengarahkan mereka ke URL ini untuk menyelesaikan proses masuk. Kemudian Salesforce mengarahkan pengguna ke URL callback yang telah Anda kaitkan dengan klien aplikasi Anda.

```
https://mydomain.us-east-1.amazoncognito.com/authorize?  
response_type=code&client_id=<your_client_id>&redirect_uri=https://  
www.example.com&identity_provider=CorpSalesforce
```

- c. Aktifkan pengaturan OAuth dan masukkan URL `/oauth2/idpresponse` titik akhir untuk domain kumpulan pengguna Anda di URL Callback. Ini adalah URL tempat Salesforce mengeluarkan kode otorisasi yang ditukar Amazon Cognito dengan token OAuth.

```
https://mydomain.us-east-1.amazoncognito.com/oauth2/idpresponse
```

9. Pilih [cakupan](#) Anda. Anda harus menyertakan openid lingkup. Untuk memberikan akses ke email dan [klaim](#) email\_verified, tambahkan cakupan email. Pisahkan cakupan berdasarkan spasi.
10. Pilih Buat.

Dalam Salesforce, ID klien disebut Kunci Konsumen, dan rahasia klien adalah Rahasia Konsumen. Perhatikan ID klien dan rahasia klien Anda. Anda akan menggunakannya di bagian selanjutnya.

## Langkah 2: Tambahkan OIDC IdP ke kumpulan pengguna Anda

Pada bagian ini, Anda mengonfigurasi kolom pengguna Anda untuk memproses permintaan autentikasi berbasis OIDC dari IdP OIDC.

Untuk menambahkan IdP OIDC (konsol Amazon Cognito)

### Tambahkan OIDC IdP


1. Masuk ke [Konsol Amazon Cognito](#). Jika diminta, masukkan AWS kredensial Anda.
2. Pilih User Pools dari menu navigasi.
3. Pilih kolom pengguna yang ada dari daftar, atau [buat kolom pengguna](#).
4. Pilih tab Pengalaman masuk. Cari Masuk Federasi dan pilih Tambahkan penyedia identitas.
5. Pilih OpenID Connect IDP.
6. Masukkan nama unik ke dalam nama Penyedia.
7. Masukkan ID klien yang Anda terima dari penyedia Anda ke ID Klien.
8. Masukkan rahasia klien yang Anda terima dari penyedia Anda ke dalam rahasia Klien.
9. Masukkan cakupan resmi untuk penyedia ini. Cakupan menentukan grup atribut pengguna mana (seperti name dan email) yang akan diminta aplikasi Anda dari penyedia Anda. Lingkup harus dipisahkan oleh spasi, mengikuti spesifikasi [OAuth 2.0](#).

Pengguna Anda akan diminta untuk menyetujui untuk memberikan atribut ini ke aplikasi Anda.

10. Pilih metode permintaan Atribut untuk menyediakan Amazon Cognito dengan metode HTTP (baik GET atau POST) yang harus digunakan untuk mengambil detail pengguna dari titik akhir UserInfo yang dioperasikan oleh penyedia Anda.
11. Pilih metode Pengaturan untuk mengambil titik akhir OpenID Connect baik dengan pengisian Otomatis melalui URL penerbit atau input Manual. Gunakan Isi otomatis melalui URL penerbit saat penyedia Anda memiliki .well-known/openid-configuration titik akhir publik tempat

Amazon Cognito dapat mengambil URL,,, dan titik akhir `authorization`. token `userInfo` `jwtks_uri`

12. Masukkan URL penerbit atau `authorization`, token `userInfo`, dan URL `jwtks_uri` endpoint dari IDP Anda.

 Note

URL harus dimulai dengan `https://`, dan tidak boleh diakhiri dengan garis miring `/`. Hanya nomor port 443 dan 80 yang dapat digunakan dengan URL ini. Misalnya, Salesforce menggunakan URL ini:

```
https://login.salesforce.com
```

Jika Anda memilih pengisian otomatis, dokumen penemuan harus menggunakan HTTPS untuk nilai

berikut: `authorization_endpoint`, `token_endpoint`, `userinfo_endpoint`, dan `jwtks_uri`. Jika tidak, login akan gagal.

13. Sub klaim OIDC dipetakan ke atribut kumpulan pengguna `Username` secara default. Anda dapat memetakan [klaim](#) OIDC lainnya ke atribut kolom pengguna. Masukkan klaim OIDC, dan pilih atribut kumpulan pengguna yang sesuai dari daftar drop-down. Misalnya, email klaim sering dipetakan ke atribut kolom pengguna `Email`.
14. Petakan atribut dari iDP Anda ke kumpulan pengguna Anda. Untuk informasi selengkapnya, lihat [Menentukan Pemetaan Atribut Penyedia Identitas untuk Kolam Pengguna Anda](#).
15. Pilih `Buat`.
16. Dari tab `Integrasi klien aplikasi`, pilih salah satu klien Aplikasi dalam daftar dan `Edit` pengaturan UI yang dihosting. Tambahkan OIDC iDP baru ke klien aplikasi di bawah penyedia Identity.
17. Pilih `Simpan` perubahan.

Untuk menambahkan IdP OIDC (AWS CLI)

- Lihat deskripsi parameter untuk metode [CreateIdentityProvider](#) API.

```
aws cognito-idp create-identity-provider
--user-pool-id string
--provider-name string
--provider-type OIDC
```

```
--provider-details map  
  
--attribute-mapping string  
--idp-identifiers (list)  
--cli-input-json string  
--generate-cli-skeleton string
```

Gunakan peta detail penyedia ini:

```
{  
  "client_id": "string",  
  "client_secret": "string",  
  "authorize_scopes": "string",  
  "attributes_request_method": "string",  
  "oidc_issuer": "string",  
  
  "authorize_url": "string",  
  "token_url": "string",  
  "attributes_url": "string",  
  "jwks_uri": "string"  
}
```

### Langkah 3: Uji konfigurasi IDP OIDC Anda

Anda dapat membuat URL otorisasi dengan menggunakan elemen dari dua bagian sebelumnya, dan menggunakannya untuk menguji konfigurasi IdP OIDC Anda.

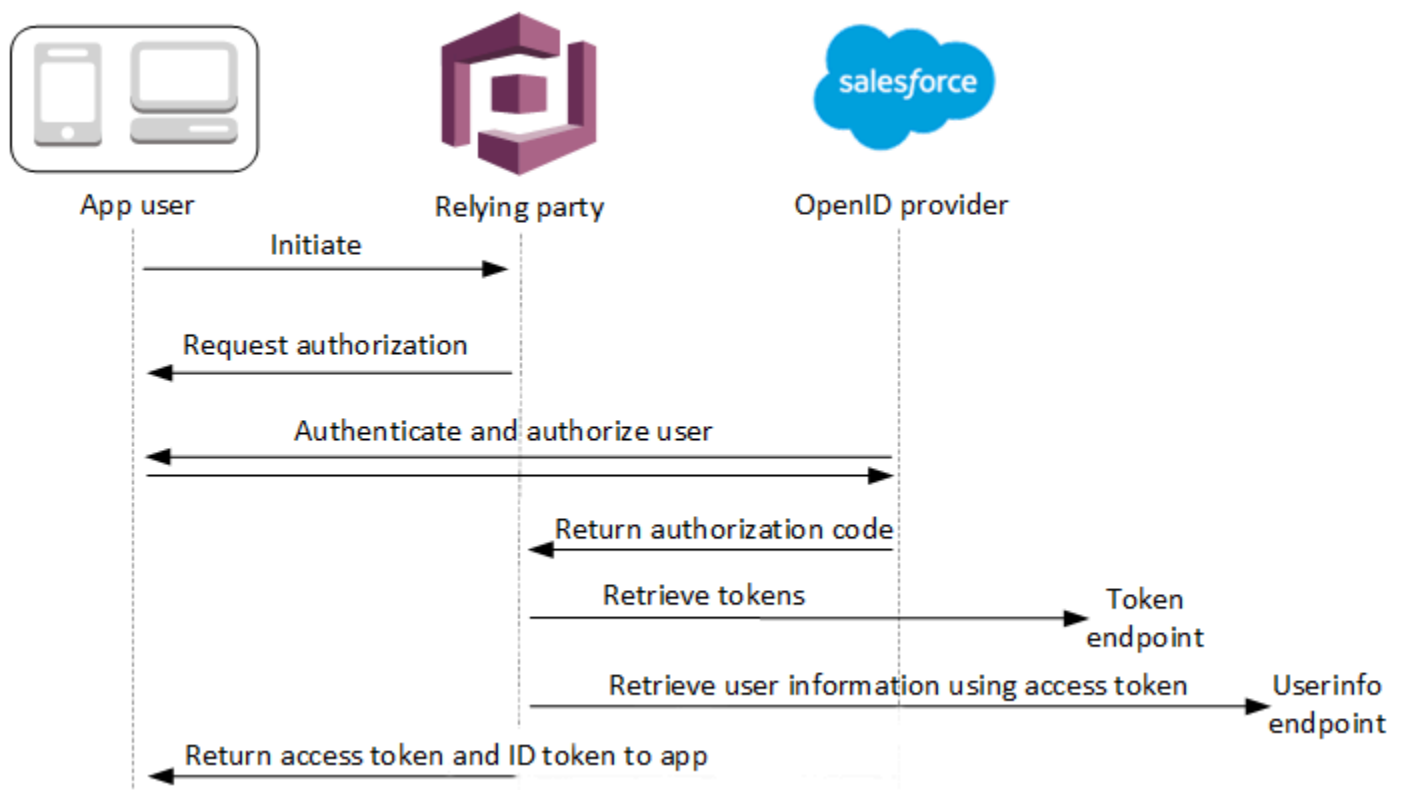
```
https://mydomain.us-east-1.amazoncognito.com/oauth2/authorize?  
response_type=code&client_id=1example23456789&redirect_uri=https://www.example.com
```

Anda dapat menemukan domain Anda di halaman konsol Nama Domain kolam pengguna. `client_id` ada di halaman Pengaturan umum. Gunakan URL panggilan balik Anda untuk parameter `redirect_uri`. Ini adalah URL halaman tempat pengguna Anda akan diarahkan setelah autentikasi berhasil.

## Aliran otentikasi IDP kumpulan pengguna OIDC

Saat pengguna Anda masuk ke aplikasi Anda menggunakan OIDC iDP, mereka melewati alur autentikasi berikut.

1. Pengguna Anda mendarat di halaman masuk bawaan Amazon Cognito, dan ditawarkan opsi untuk masuk melalui IdP OIDC seperti Salesforce.
2. Pengguna Anda dialihkan ke `authorization` titik akhir OIDC iDP.
3. Setelah pengguna Anda diautentikasi, IdP OIDC mengalihkan ke Amazon Cognito dengan kode otorisasi.
4. Amazon Cognito menukar kode otorisasi dengan IdP OIDC untuk sebuah token akses.
5. Amazon Cognito membuat atau memperbarui akun pengguna di kolam pengguna Anda.
6. Amazon Cognito menerbitkan token pembawa aplikasi Anda, yang mungkin termasuk identitas, akses, dan token penyegaran.



**Note**

Amazon Cognito membatalkan permintaan otentikasi yang tidak selesai dalam 5 menit, dan mengarahkan pengguna ke UI yang dihosting. Halaman menampilkan pesan `Something went wrong` kesalahan.

OIDC adalah lapisan identitas di atas OAuth 2.0, yang menentukan token identitas berformat JSON (JWT) yang dikeluarkan oleh aplikasi klien OIDC (pihak yang mengandalkan). Lihat dokumentasi untuk IdP OIDC Anda untuk informasi tentang menambahkan Amazon Cognito sebagai pihak yang mengandalkan OIDC.

Saat pengguna mengautentikasi dengan pemberian kode otorisasi, kumpulan pengguna mengembalikan ID, akses, dan token penyegaran. Token ID adalah token [OIDC](#) standar untuk manajemen identitas, dan token akses adalah token [OAuth 2.0](#) standar. Untuk informasi selengkapnya tentang jenis hibah yang dapat didukung oleh klien aplikasi kumpulan pengguna Anda, lihat [Otorisasi titik akhir](#).

Bagaimana kumpulan pengguna memproses klaim dari penyedia OIDC

Saat pengguna Anda menyelesaikan proses masuk dengan penyedia OIDC pihak ketiga, UI yang dihosting Amazon Cognito akan mengambil kode otorisasi dari iDP. Kumpulan pengguna Anda menukar kode otorisasi untuk akses dan token ID dengan token titik akhir IDP Anda. Kumpulan pengguna Anda tidak meneruskan token ini ke pengguna atau aplikasi Anda, tetapi menggunakannya untuk membuat profil pengguna dengan data yang ditampilkan dalam klaim dalam tokennya sendiri.

Amazon Cognito tidak secara independen memvalidasi token akses. Sebagai gantinya, ia meminta informasi atribut pengguna dari `userInfo` titik akhir penyedia dan mengharapkan permintaan ditolak jika token tidak valid.

Amazon Cognito memvalidasi token ID penyedia dengan pemeriksaan berikut:

1. Periksa apakah penyedia menandatangani token dengan algoritma dari set berikut: RSA, HMAC, Elliptic Curve.
2. Jika penyedia menandatangani token dengan algoritme penandatanganan asimetris, periksa apakah ID kunci penandatanganan dalam `kid` klaim token terdaftar di titik `jwtks_uri` akhir penyedia.
3. Bandingkan tanda tangan token ID dengan tanda tangan yang diharapkan berdasarkan metadata penyedia.



4. Bandingkan `iss` klaim dengan penerbit OIDC yang dikonfigurasi untuk IDP.
5. Bandingkan `aud` klaim yang cocok dengan ID klien yang dikonfigurasi di IDP, atau berisi ID klien yang dikonfigurasi jika ada beberapa nilai dalam klaim. `aud`
6. Periksa apakah stempel waktu dalam `exp` klaim tidak sebelum waktu saat ini.

Kumpulan pengguna Anda memvalidasi token ID, lalu mencoba permintaan ke `userInfo` titik akhir penyedia dengan token akses penyedia. Ini mengambil informasi profil pengguna yang cakupan dalam token akses mengotorisasi untuk dibaca. Kumpulan pengguna Anda kemudian mencari atribut pengguna yang telah Anda tetapkan seperti yang diperlukan dalam kumpulan pengguna Anda. Anda harus membuat pemetaan atribut dalam konfigurasi penyedia Anda untuk atribut yang diperlukan. Kumpulan pengguna Anda memeriksa token ID penyedia dan `userInfo` responsnya. Kumpulan pengguna Anda menulis semua klaim yang mencocokkan aturan pemetaan dengan atribut pengguna di profil pengguna kumpulan pengguna. Kumpulan pengguna Anda mengabaikan atribut yang cocok dengan aturan pemetaan tetapi tidak diperlukan dan tidak ditemukan dalam klaim penyedia.

## Menentukan pemetaan atribut penyedia identitas untuk kumpulan pengguna Anda

Anda dapat menggunakan AWS Management Console, AWS CLI atau API, untuk menentukan pemetaan atribut untuk penyedia identitas (IDP) kumpulan pengguna Anda.

### Hal yang perlu diketahui tentang pemetaan

Sebelum Anda mulai mengatur pemetaan atribut pengguna, tinjau detail penting berikut.

- Saat pengguna federasi masuk ke aplikasi Anda, pemetaan harus ada untuk setiap atribut kumpulan pengguna yang diperlukan oleh kumpulan pengguna Anda. Misalnya, jika kumpulan pengguna Anda memerlukan `email` atribut untuk mendaftar, petakan atribut ini ke ekuivalennya dari IDP.
- Secara default, alamat email yang dipetakan tidak diverifikasi. Anda tidak dapat memverifikasi alamat email yang dipetakan menggunakan kode satu kali. Sebagai gantinya, petakan atribut dari IDP Anda untuk mendapatkan status verifikasi. Misalnya, Google dan sebagian besar penyedia OIDC menyertakan atribut `email_verified`.
- Anda dapat memetakan token penyedia identitas (IDP) ke atribut kustom di kumpulan pengguna Anda. Penyedia sosial menyajikan token akses, dan penyedia OIDC menyajikan akses dan token ID. Untuk memetakan token, tambahkan atribut kustom dengan panjang maksimum 2.048 karakter,

berikan akses tulis klien aplikasi Anda ke atribut, dan petakan `access_token` atau `id_token` dari iDP ke atribut kustom.

- Untuk setiap atribut kumpulan pengguna yang dipetakan, panjang nilai maksimum 2.048 karakter harus cukup besar untuk nilai yang diperoleh Amazon Cognito dari iDP. Jika tidak, Amazon Cognito melaporkan kesalahan saat pengguna masuk ke aplikasi Anda. Amazon Cognito tidak mendukung pemetaan token iDP ke atribut khusus ketika token memiliki panjang lebih dari 2.048 karakter.
- Amazon Cognito memperoleh `username` atribut dalam profil pengguna federasi dari klaim spesifik yang dilewati oleh IDP federasi Anda, seperti yang ditunjukkan pada tabel berikut. Amazon Cognito menambahkan nilai atribut ini dengan nama iDP Anda, misalnya, `MyOIDCIdP_[sub]`. Jika Anda ingin pengguna federasi memiliki atribut yang sama persis dengan atribut di direktori pengguna eksternal, petakan atribut tersebut ke atribut login Amazon Cognito seperti `preferred_username`.

Penyedia Identitas	<code>username</code> atribut sumber
Facebook	<code>id</code>
Google	<code>sub</code>
Login with Amazon	<code>user_id</code>
Masuk dengan Apple	<code>sub</code>
Penyedia SAMP	<code>NameID</code>
Penyedia OpenID Connect (OIDC)	<code>sub</code>

- Amazon Cognito harus dapat memperbarui atribut kolom pengguna yang dipetakan ketika pengguna masuk ke aplikasi Anda. Saat pengguna masuk melalui iDP, Amazon Cognito memperbarui atribut yang dipetakan dengan informasi terbaru dari iDP. Amazon Cognito memperbarui setiap atribut yang dipetakan, bahkan jika nilainya saat ini sudah sesuai dengan informasi terbaru. Untuk memastikan bahwa Amazon Cognito dapat memperbarui atribut, periksa persyaratan berikut:
  - Semua atribut kustom kumpulan pengguna yang Anda petakan dari IDP Anda harus bisa berubah. Anda dapat memperbarui atribut kustom yang bisa berubah kapan saja. Sebaliknya, Anda hanya dapat menetapkan nilai untuk atribut kustom yang tidak dapat diubah pengguna saat pertama kali membuat profil pengguna. Untuk membuat atribut kustom yang bisa berubah di konsol Amazon Cognito, aktifkan kotak centang `Mutable` untuk atribut yang Anda tambahkan

saat memilih Tambahkan atribut kustom di tab Pengalaman pendaftaran. Atau, jika Anda membuat kumpulan pengguna menggunakan operasi [CreateUserPoolAPI](#), Anda dapat mengatur Mutable parameter untuk masing-masing atribut `inittrue`. Jika idP Anda mengirimkan nilai untuk atribut immutable yang dipetakan, Amazon Cognito mengembalikan kesalahan dan proses masuk gagal.

- Dalam pengaturan klien aplikasi untuk aplikasi Anda, atribut yang dipetakan harus dapat ditulis. Anda dapat mengatur atribut mana yang dapat ditulis di halaman Klien aplikasi di konsol Amazon Cognito. Atau, jika Anda membuat klien aplikasi dengan menggunakan operasi API [CreateUserPoolClient](#), Anda dapat menambahkan atribut ini ke array `WriteAttributes`. Jika idP Anda mengirimkan nilai untuk atribut non-writable yang dipetakan, Amazon Cognito tidak menyetel nilai atribut dan melanjutkan dengan autentikasi.
- Ketika atribut IDP berisi beberapa nilai, Amazon Cognito meratakan semua nilai menjadi string tunggal yang dibatasi koma dan formulir URL-mengkodekan nilai yang mengandung karakter non-alfanumerik (tidak termasuk karakter `"`, `"`, `'`, dan `'`). `. - * _` Anda harus memecahkan kode dan mengurai nilai individual sebelum menggunakannya di aplikasi.

## Menentukan pemetaan atribut penyedia identitas untuk kumpulan pengguna Anda (AWS Management Console)

Anda dapat menggunakan AWS Management Console untuk menentukan pemetaan atribut untuk idP kumpulan pengguna Anda.

### Note

Amazon Cognito akan memetakan klaim masuk ke atribut kolam pengguna hanya jika klaim ada di token masuk. Jika klaim yang dipetakan sebelumnya tidak lagi ada di token masuk, klaim tersebut tidak akan dihapus atau diubah. Jika aplikasi Anda memerlukan pemetaan klaim yang dihapus, Anda dapat menggunakan pemicu Lambda Pra-Authentikasi untuk menghapus atribut kustom selama autentikasi dan mengizinkan atribut ini terisi kembali dari token yang masuk.

Untuk menentukan pemetaan atribut idP sosial

1. Masuk ke [konsol Amazon Cognito](#). Jika diminta, masukkan AWS kredensial Anda.
2. Di panel navigasi, pilih Kumpulan Pengguna, dan pilih kumpulan pengguna yang ingin Anda edit.

3. Pilih tab Pengalaman masuk dan temukan Masuk Federasi.
4. Pilih Tambahkan penyedia identitas, atau pilih Facebook, Google, Amazon, atau Apple iDP yang telah Anda konfigurasi. Temukan pemetaan Atribut dan pilih Edit.

Untuk informasi selengkapnya tentang menambahkan IdP sosial, lihat. [Menggunakan penyedia identitas sosial dengan kumpulan pengguna](#)

5. Untuk setiap atribut yang perlu Anda petakan, selesaikan langkah-langkah berikut:
  - a. Pilih atribut dari kolom atribut User pool. Ini adalah atribut yang ditetapkan ke profil pengguna di kumpulan pengguna Anda. Atribut kustom terdaftar setelah atribut standar.
  - b. Pilih atribut dari <provider>kolom atribut. Ini akan menjadi atribut yang diteruskan dari direktori penyedia. Atribut yang dikenal dari penyedia sosial disediakan dalam daftar drop-down.
  - c. Untuk memetakan atribut tambahan antara IDP dan Amazon Cognito, pilih Tambahkan atribut lain.
6. Pilih Simpan perubahan.

Untuk menentukan pemetaan atribut penyedia SAML

1. Masuk ke [Konsol Amazon Cognito](#). Jika diminta, masukkan AWS kredensial Anda.
2. Di panel navigasi, pilih Kumpulan Pengguna, dan pilih kumpulan pengguna yang ingin Anda edit.
3. Pilih tab Pengalaman masuk dan temukan Masuk Federasi.
4. Pilih Tambahkan penyedia identitas, atau pilih IDP SAMP yang telah Anda konfigurasi. Temukan pemetaan Atribut, dan pilih Edit. Untuk informasi selengkapnya tentang menambahkan IDP SAMP, lihat. [Menggunakan penyedia identitas SAMP dengan kumpulan pengguna](#)
5. Untuk setiap atribut yang perlu Anda petakan, selesaikan langkah-langkah berikut:
  - a. Pilih atribut dari kolom atribut User pool. Ini adalah atribut yang ditetapkan ke profil pengguna di kumpulan pengguna Anda. Atribut kustom terdaftar setelah atribut standar.
  - b. Pilih atribut dari kolom atribut SAMP. Ini akan menjadi atribut yang diteruskan dari direktori penyedia.

IDP Anda mungkin menawarkan contoh pernyataan SAMP untuk referensi. Beberapa IdPs menggunakan nama sederhana, seperti `email`, sementara yang lain menggunakan nama atribut berformat URL yang mirip dengan:

```
http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress
```

- c. Untuk memetakan atribut tambahan antara IDP dan Amazon Cognito, pilih Tambahkan atribut lain.
6. Pilih Simpan perubahan.

## Menentukan pemetaan atribut penyedia identitas untuk kumpulan pengguna Anda (AWS CLI dan API) AWS

Gunakan perintah berikut untuk menentukan pemetaan atribut IDP untuk kumpulan pengguna Anda.

Untuk menentukan pemetaan atribut pada waktu pembuatan penyedia

- AWS CLI: `aws cognito-idp create-identity-provider`

Contoh dengan file metadata: `aws cognito-idp create-identity-provider --user-pool-id <user_pool_id> --provider-name=SAML_provider_1 --provider-type SAML --provider-details file:///details.json --attribute-mapping email=http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress`

Di mana `details.json` berisi:

```
{
  "MetadataFile": "<SAML metadata XML>"
}
```

### Note

Jika `<SAML metadata XML>` berisi kutipan (") apa pun, mereka harus diloloskan (\").

Contoh dengan URL metadata:

```
aws cognito-idp create-identity-provider \
--user-pool-id us-east-1_EXAMPLE \
--provider-name=SAML_provider_1 \
--provider-type SAML \
--provider-details MetadataURL=https://myidp.example.com/saml/metadata \
```

```
--attribute-mapping email=http://schemas.xmlsoap.org/ws/2005/05/identity/claims/  
emailaddress
```

- AWS API: [CreateIdentityProvider](#)

Untuk menentukan pemetaan atribut untuk IDP yang ada

- AWS CLI: `aws cognito-idp update-identity-provider`

```
Contoh: aws cognito-idp update-identity-provider --user-pool-id  
<user_pool_id> --provider-name <provider_name> --attribute-mapping  
email=http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress
```

- AWS API: [UpdateIdentityProvider](#)

Untuk mendapatkan informasi tentang pemetaan atribut untuk IDP tertentu

- AWS CLI: `aws cognito-idp describe-identity-provider`

```
Contoh: aws cognito-idp describe-identity-provider --user-pool-id  
<user_pool_id> --provider-name <provider_name>
```

- AWS API: [DescribeIdentityProvider](#)

## Menautkan pengguna gabungan ke profil pengguna yang ada

Seringkali, pengguna yang sama memiliki profil dengan beberapa penyedia identitas (IdPs) yang telah Anda sambungkan ke kumpulan pengguna Anda. Amazon Cognito dapat menautkan setiap kemunculan pengguna ke profil pengguna yang sama di direktori Anda. Dengan cara ini, satu orang yang memiliki beberapa pengguna IDP dapat memiliki pengalaman yang konsisten di aplikasi Anda. [AdminLinkProviderForUser](#) memberi tahu Amazon Cognito untuk mengenali ID unik pengguna di direktori federasi Anda sebagai pengguna di kumpulan pengguna. Pengguna di kumpulan pengguna Anda dihitung sebagai satu pengguna aktif bulanan (MAU) untuk tujuan [penagihan](#) ketika Anda memiliki nol atau lebih identitas gabungan yang terkait dengan profil pengguna.

Saat pengguna federasi masuk ke kumpulan pengguna Anda untuk pertama kalinya, Amazon Cognito mencari profil lokal yang telah Anda tautkan ke identitas mereka. Jika tidak ada profil tertaut, kumpulan pengguna Anda akan membuat profil baru. Anda dapat membuat profil lokal dan menautkannya ke pengguna federasi kapan saja sebelum login pertama mereka, dalam permintaan `AdminLinkProviderForUser` API, baik dalam tugas prestaging yang direncanakan atau di file.

[Pemicu Lambda pra pendaftaran](#) Setelah pengguna masuk dan Amazon Cognito mendeteksi profil lokal yang ditautkan, kumpulan pengguna akan membaca klaim pengguna dan membandingkannya dengan aturan pemetaan untuk IDP. Kumpulan pengguna Anda kemudian memperbarui profil lokal yang ditautkan dengan klaim yang dipetakan dari proses masuk mereka. Dengan cara ini, Anda dapat mengonfigurasi profil lokal dengan klaim akses dan menyimpan klaim identitas mereka up-to-date dengan penyedia Anda. Setelah Amazon Cognito mencocokkan pengguna federasi Anda dengan profil yang ditautkan, mereka selalu masuk ke profil tersebut. Anda kemudian dapat menautkan lebih banyak identitas penyedia pengguna Anda ke profil yang sama, memberikan satu pelanggan pengalaman yang konsisten di aplikasi Anda. Untuk menautkan pengguna federasi yang sebelumnya telah masuk, Anda harus terlebih dahulu menghapus profil mereka yang ada. Anda dapat mengidentifikasi profil yang ada berdasarkan formatnya: `[Provider name]_identifier`. Misalnya, `LoginWithAmazon_amzn1.account.AFAEXAMPLE`. Pengguna yang Anda buat dan kemudian ditautkan ke identitas pengguna pihak ketiga memiliki nama pengguna yang mereka buat, dan `identities` atribut yang berisi detail identitas tertaut mereka.

#### Important

Karena `AdminLinkProviderForUser` memungkinkan pengguna dengan identitas federasi eksternal untuk masuk sebagai pengguna yang ada di kumpulan pengguna, sangat penting bahwa itu hanya digunakan dengan atribut eksternal IDPs dan penyedia yang telah dipercaya oleh pemilik aplikasi.

Misalnya, jika Anda adalah penyedia layanan terkelola (MSP) dengan aplikasi yang Anda bagikan dengan beberapa pelanggan. Setiap pelanggan masuk ke aplikasi Anda melalui Active Directory Federation Services (ADFS). Administrator TI Anda, Carlos, memiliki akun di setiap domain pelanggan Anda. Anda ingin Carlos diakui sebagai administrator aplikasi setiap kali dia masuk, terlepas dari IDP.

ADFS Anda IDPs menyajikan alamat email `Carlos msp_carlos@example.com` dalam `email` klaim pernyataan SAMP Carlos ke Amazon Cognito. Anda membuat pengguna di kumpulan pengguna Anda dengan nama pengguna `Carlos`. Perintah berikut AWS Command Line Interface (AWS CLI) menghubungkan identitas Carlos dari IDPs ADFS1, ADFS2, dan ADFS3.

#### Note

Anda dapat menautkan pengguna berdasarkan klaim atribut tertentu. Kemampuan ini unik untuk OIDC dan SAMP. IDPs Untuk jenis penyedia lainnya, Anda harus

menautkan berdasarkan atribut sumber tetap. Untuk informasi lebih lanjut, lihat [AdminLinkProviderForUser](#). Anda harus mengatur `ProviderAttributeName` `Cognito_Subject` kapan Anda menautkan iDP sosial ke profil pengguna. `ProviderAttributeValue` harus menjadi pengenal unik pengguna dengan IDP Anda.

```
aws cognito-idp admin-link-provider-for-user \
--user-pool-id us-east-1_EXAMPLE \
--destination-user ProviderAttributeValue=Carlos,ProviderName=Cognito \
--source-user
  ProviderName=ADFS1,ProviderAttributeName=email,ProviderAttributeValue=msp_carlos@example.com

aws cognito-idp admin-link-provider-for-user \
--user-pool-id us-east-1_EXAMPLE \
--destination-user ProviderAttributeValue=Carlos,ProviderName=Cognito \
--source-user
  ProviderName=ADFS2,ProviderAttributeName=email,ProviderAttributeValue=msp_carlos@example.com

aws cognito-idp admin-link-provider-for-user \
--user-pool-id us-east-1_EXAMPLE \
--destination-user ProviderAttributeValue=Carlos,ProviderName=Cognito \
--source-user
  ProviderName=ADFS3,ProviderAttributeName=email,ProviderAttributeValue=msp_carlos@example.com
```

Profil pengguna Carlos di kumpulan pengguna Anda sekarang memiliki identities atribut berikut.

```
[{
  "userId": "msp_carlos@example.com",
  "providerName": "ADFS1",
  "providerType": "SAML",
  "issuer": "http://auth.example.com",
  "primary": false,
  "dateCreated": 1111111111111111
}, {
  "userId": "msp_carlos@example.com",
  "providerName": "ADFS2",
  "providerType": "SAML",
  "issuer": "http://auth2.example.com",
  "primary": false,
  "dateCreated": 1111111111111111
```



```
}, {
  "userId": "msp_carlos@example.com",
  "providerName": "ADFS3",
  "providerType": "SAML",
  "issuer": "http://auth3.example.com",
  "primary": false,
  "dateCreated": 1111111111111111
}]
```

## Hal-hal yang perlu diketahui tentang menautkan pengguna federasi

- Anda dapat menautkan hingga lima pengguna federasi ke setiap profil pengguna.
- Anda dapat menautkan pengguna federasi ke profil pengguna federasi yang ada, atau ke pengguna lokal.
- Anda tidak dapat menautkan penyedia ke profil pengguna di AWS Management Console.
- Token ID pengguna Anda berisi semua penyedia terkait mereka dalam `identities` klaim.
- Anda dapat menyetel kata sandi untuk profil pengguna federasi yang dibuat secara otomatis dalam permintaan API. [AdminSetUserPassword](#) Status pengguna itu kemudian berubah dari `EXTERNAL_PROVIDER` menjadi `CONFIRMED`. Pengguna dalam status ini dapat masuk sebagai pengguna federasi, dan memulai alur otentikasi di API seperti pengguna lokal yang ditautkan. Mereka juga dapat memodifikasi kata sandi dan atribut mereka dalam permintaan API yang diautentikasi token seperti dan. [ChangePasswordUpdateUserAttributes](#) Sebagai praktik keamanan terbaik dan agar pengguna tetap sinkron dengan iDP eksternal Anda, jangan setel kata sandi pada profil pengguna gabungan. Sebagai gantinya, tautkan pengguna ke profil lokal dengan `AdminLinkProviderForUser`.
- Amazon Cognito mengisi atribut pengguna ke profil pengguna lokal yang ditautkan saat pengguna masuk melalui idP mereka. Amazon Cognito memproses klaim identitas dalam token ID dari IDP OIDC, dan juga memeriksa titik akhir penyedia OAuth 2.0 dan `userInfo` OIDC. Amazon Cognito memprioritaskan informasi dalam token ID daripada informasi dari `userInfo`

Ketika Anda mengetahui bahwa pengguna Anda tidak lagi menggunakan akun pengguna eksternal yang telah Anda tautkan ke profil mereka, Anda dapat memisahkan akun pengguna tersebut dengan pengguna kumpulan pengguna Anda. Saat menautkan pengguna, Anda memberikan nama atribut, nilai atribut, dan nama penyedia pengguna dalam permintaan. Untuk menghapus profil yang tidak lagi dibutuhkan pengguna Anda, buat permintaan [AdminDisableProviderForUser](#) API dengan parameter yang setara.

Lihat [AdminLinkProviderForUsers](#) sintaks perintah tambahan dan contoh di AWS SDK.

## Menyesuaikan alur kerja kumpulan pengguna dengan pemicu Lambda

Amazon Cognito berfungsi dengan AWS Lambda fungsi untuk mengubah perilaku otentikasi kumpulan pengguna Anda. Anda dapat mengonfigurasi kumpulan pengguna untuk menjalankan fungsi Lambda secara otomatis sebelum pendaftaran pertama mereka, setelah menyelesaikan otentikasi, dan pada beberapa tahap di antaranya. Fungsi Anda dapat mengubah perilaku default alur autentikasi Anda, membuat permintaan API untuk memodifikasi kumpulan pengguna Anda atau AWS sumber daya lainnya, dan berkomunikasi dengan sistem eksternal. Kode dalam fungsi Lambda Anda adalah milik Anda sendiri. Amazon Cognito mengirimkan data peristiwa ke fungsi Anda, menunggu fungsi memproses data, dan dalam kebanyakan kasus mengantisipasi peristiwa respons yang mencerminkan perubahan apa pun yang ingin Anda lakukan pada sesi.

Dalam sistem peristiwa permintaan dan respons, Anda dapat memperkenalkan tantangan otentikasi Anda sendiri, memigrasikan pengguna antara kumpulan pengguna Anda dan toko identitas lain, menyesuaikan pesan, dan memodifikasi token web JSON (JWT).

Pemicu Lambda dapat menyesuaikan respons yang diberikan Amazon Cognito kepada pengguna Anda setelah mereka memulai tindakan di kumpulan pengguna Anda. Misalnya, Anda dapat mencegah login oleh pengguna yang jika tidak akan berhasil. Mereka juga dapat melakukan operasi runtime terhadap AWS lingkungan Anda, API eksternal, database, atau penyimpanan identitas. Pemicu pengguna yang bermigrasi, misalnya, dapat menggabungkan tindakan eksternal dengan perubahan di Amazon Cognito: Anda dapat mencari informasi pengguna di direktori eksternal, lalu mengatur atribut pada pengguna baru berdasarkan informasi eksternal tersebut.

Saat Anda memiliki pemicu Lambda yang ditetapkan ke kumpulan pengguna, Amazon Cognito akan mengganggu alur defaultnya untuk meminta informasi dari fungsi Anda. Amazon Cognito menghasilkan acara JSON dan meneruskannya ke fungsi Anda. Acara ini berisi informasi tentang permintaan pengguna Anda untuk membuat akun pengguna, masuk, mengatur ulang kata sandi, atau memperbarui atribut. Fungsi Anda kemudian memiliki kesempatan untuk mengambil tindakan, atau mengirim acara kembali tanpa dimodifikasi.

Tabel berikut merangkum beberapa cara Anda dapat menggunakan pemicu Lambda untuk menyesuaikan operasi kumpulan pengguna:

Alur Kolam pengguna	Operasi	Deskripsi
Alur Otentikasi Kustom	Tentukan Tantangan Auth	Menentukan tantangan berikutnya dalam alur autentikasi kustom
	Buat Tantangan Auth	Membuat tantangan dalam alur autentikasi kustom
	Verifikasi Respon Tantangan Auth	Menentukan apakah respon benar dalam aliran autentikasi kustom
Acara Otentikasi	<a href="#">the section called “Pemicu Lambda pra otentikasi”</a>	Validasi kustom untuk menerima atau menolak permintaan masuk
	<a href="#">the section called “Posting otentikasi pemicu Lambda”</a>	Log peristiwa untuk analisis kustom
	<a href="#">the section called “Pemicu Lambda generasi pra token”</a>	Menambah atau menekan klaim token
Mendaftar	<a href="#">the section called “Pemicu Lambda pra pendaftaran”</a>	Melakukan validasi kustom yang menerima atau menolak permintaan pendaftaran
	<a href="#">the section called “Posting konfirmasi pemicu Lambda”</a>	Menambahkan pesan selamat datang khusus atau pencatatan peristiwa untuk analitik kustom
	<a href="#">the section called “Memigrasi pengguna pemicu Lambda”</a>	Memigrasi pengguna dari direktori pengguna yang ada ke kumpulan pengguna
Pesan	<a href="#">the section called “Pesan khusus Lambda pemicu”</a>	Melakukan kustomisasi lanjutan dan lokalisasi pesan

Alur Kolam pengguna	Operasi	Deskripsi
Pembuatan Token	<a href="#">the section called “Pemicu Lambda generasi pra token”</a>	Menambahkan atau menghapus atribut dalam token Id
Penyedia email dan SMS pihak ketiga	<a href="#">the section called “Pemicu Lambda pengirim kustom”</a>	Menggunakan penyedia pihak ketiga untuk mengirim pesan SMS dan email

## Topik

- [Pertimbangan penting](#)
- [Menambahkan pemicu Lambda kumpulan pengguna](#)
- [Acara pemicu Lambda kumpulan pengguna](#)
- [Kumpulan pengguna Lambda memicu parameter umum](#)
- [Menghubungkan operasi API ke pemicu Lambda](#)
- [Menghubungkan pemicu Lambda ke operasi fungsional kumpulan pengguna](#)
- [Pemicu Lambda pra pendaftaran](#)
- [Posting konfirmasi pemicu Lambda](#)
- [Pemicu Lambda pra otentikasi](#)
- [Posting otentikasi pemicu Lambda](#)
- [Tantangan otentikasi kustom pemicu Lambda](#)
- [Pemicu Lambda generasi pra token](#)
- [Memigrasi pengguna pemicu Lambda](#)
- [Pesan khusus Lambda pemicu](#)
- [Pemicu Lambda pengirim kustom](#)

## Pertimbangan penting

Saat Anda menyiapkan kumpulan pengguna untuk fungsi Lambda, pertimbangkan hal berikut:

- Peristiwa yang dikirimkan Amazon Cognito ke pemicu Lambda Anda mungkin berubah dengan fitur baru. Posisi elemen respons dan permintaan dalam hierarki JSON mungkin berubah, atau

nama elemen dapat ditambahkan. Dalam fungsi Lambda Anda, Anda dapat mengharapkan untuk menerima pasangan nilai kunci elemen input yang dijelaskan dalam panduan ini, tetapi validasi input yang lebih ketat dapat menyebabkan fungsi Anda gagal.

- Anda dapat memilih salah satu dari beberapa versi acara yang dikirimkan Amazon Cognito ke beberapa pemicu. Beberapa versi mungkin mengharuskan Anda untuk menerima perubahan pada harga Amazon Cognito Anda. Untuk informasi selengkapnya tentang harga, lihat Harga [Amazon Cognito](#). Untuk menyesuaikan token akses di a[Pemicu Lambda generasi pra token](#), Anda harus mengonfigurasi kumpulan pengguna Anda dengan [fitur keamanan lanjutan](#) dan memperbarui konfigurasi pemicu Lambda Anda untuk menggunakan peristiwa versi 2.
- Kecuali [Pemicu Lambda pengirim kustom](#), Amazon Cognito memanggil fungsi Lambda secara serempak. Saat Amazon Cognito memanggil fungsi Lambda Anda, ia harus merespons dalam 5 detik. Jika tidak dan jika panggilan dapat dicoba lagi, Amazon Cognito mencoba kembali panggilan tersebut. Setelah tiga upaya yang gagal, fungsi tersebut habis waktu. Anda tidak dapat mengubah nilai batas waktu lima detik ini. Untuk informasi selengkapnya, lihat [Model pemrograman Lambda](#) di Panduan AWS Lambda Pengembang.

Amazon Cognito tidak mencoba lagi panggilan fungsi yang mengembalikan [kesalahan Invoke](#) dengan kode status HTTP 500-599. Kode-kode ini menunjukkan masalah konfigurasi yang membuat Lambda tidak dapat meluncurkan fungsi. Untuk informasi selengkapnya, lihat [Penanganan kesalahan dan percobaan ulang otomatis](#). AWS Lambda

- Anda tidak dapat mendeklarasikan versi fungsi dalam konfigurasi pemicu Lambda Anda. Kumpulan pengguna Amazon Cognito memanggil versi terbaru fungsi Anda secara default. Namun, Anda dapat mengaitkan versi fungsi dengan alias dan menyetel pemicu Anda LambdaArn ke alias ARN dalam permintaan atau [CreateUserPoolAPI](#). [UpdateUserPool](#) Opsi ini tidak tersedia di AWS Management Console. Untuk informasi selengkapnya tentang alias, lihat Alias [fungsi Lambda](#) di AWS Lambda Panduan Pengembang.
- Jika Anda menghapus pemicu Lambda, Anda harus memperbarui pemicu yang sesuai di kumpulan pengguna. Sebagai contoh, jika Anda menghapus pemicu pasca autentikasi, Anda harus mengatur pemicu Pasca autentikasi di kolam pengguna yang sesuai menjadi tidak ada.
- Jika fungsi Lambda Anda tidak mengembalikan parameter permintaan dan respons ke Amazon Cognito, atau menampilkan kesalahan, peristiwa autentikasi tidak berhasil. Anda dapat mengembalikan kesalahan dalam fungsi Anda untuk mencegah pendaftaran pengguna, otentikasi, pembuatan token, atau tahap lain dari aliran otentikasi mereka yang memanggil pemicu Lambda.

UI yang dihosting Amazon Cognito mengembalikan kesalahan yang dihasilkan oleh pemicu Lambda sebagai teks kesalahan di atas prompt masuk. API kumpulan pengguna Amazon Cognito

mengembalikan kesalahan pemicu dalam format. `[trigger] failed with error [error text from response]` Sebagai praktik terbaik, hanya menghasilkan kesalahan dalam fungsi Lambda Anda yang Anda ingin pengguna Anda lihat. Gunakan metode keluaran seperti `print()` mencatat informasi sensitif atau debugging apa pun ke CloudWatch Log. Sebagai contoh, lihat [Contoh pra pendaftaran: Tolak pendaftaran jika nama pengguna memiliki kurang dari lima karakter](#).

- Anda dapat menambahkan fungsi Lambda di fungsi lain Akun AWS sebagai pemicu untuk kumpulan pengguna Anda. Anda harus menambahkan pemicu lintas akun dengan operasi [CreateUserPool](#) dan [UpdateUserPool](#) API, atau padanannya di dalam dan. AWS CloudFormation AWS CLI Anda tidak dapat menambahkan fungsi lintas akun di. AWS Management Console
- Saat Anda menambahkan pemicu Lambda di konsol Amazon Cognito, Amazon Cognito menambahkan kebijakan berbasis sumber daya ke fungsi Anda yang memungkinkan kumpulan pengguna menjalankan fungsi tersebut. Saat membuat pemicu Lambda di luar konsol Amazon Cognito, termasuk fungsi lintas akun, Anda harus menambahkan izin ke kebijakan berbasis sumber daya fungsi Lambda. Izin tambahan Anda harus mengizinkan Amazon Cognito untuk menjalankan fungsi atas nama kumpulan pengguna Anda. Anda dapat [menambahkan izin dari Konsol Lambda](#) atau menggunakan operasi API Lambda [AddPermission](#).

### Contoh Kebijakan Berbasis Sumber Daya Lambda

Contoh kebijakan berbasis sumber daya Lambda berikut memberi Amazon Cognito kemampuan terbatas untuk menjalankan fungsi Lambda. Amazon Cognito hanya dapat menjalankan fungsi ketika melakukannya atas nama kumpulan pengguna dalam `aws:SourceArn` kondisi dan akun dalam kondisi tersebut. `aws:SourceAccount`

```
{
  "Version": "2012-10-17",
  "Id": "default",
  "Statement": [
    {
      "Sid": "lambda-allow-cognito",
      "Effect": "Allow",
      "Principal": {
        "Service": "cognito-idp.amazonaws.com"
      },
      "Action": "lambda:InvokeFunction",
      "Resource": "<your Lambda function ARN>",
      "Condition": {
        "StringEquals": {
          "AWS:SourceAccount": "<your account number>"
        }
      }
    }
  ]
}
```

```
    },
    "ArnLike": {
      "AWS:SourceArn": "<your user pool ARN>"
    }
  }
}
```

## Menambahkan pemicu Lambda kumpulan pengguna

Untuk menambahkan pemicu Lambda kolam pengguna dengan konsol

1. Gunakan [konsol Lambda](#) untuk membuat fungsi Lambda. Untuk informasi selengkapnya tentang fungsi Lambda, lihat [Panduan Developer AWS Lambda](#).
2. Buka [konsol Amazon Cognito](#), lalu pilih Kumpulan Pengguna.
3. Pilih kolam pengguna yang ada dari daftar, atau [buat kolam pengguna](#).
4. Pilih tab Properti kumpulan pengguna dan temukan pemicu Lambda.
5. Pilih Tambahkan pemicu Lambda.
6. Pilih Kategori pemicu Lambda berdasarkan tahap otentikasi yang ingin Anda sesuaikan.
7. Pilih Tetapkan fungsi Lambda dan pilih fungsi yang Wilayah AWS sama dengan kumpulan pengguna Anda.

### Note

Jika AWS Identity and Access Management kredensial (IAM) Anda memiliki izin untuk memperbarui fungsi Lambda, Amazon Cognito menambahkan kebijakan berbasis sumber daya Lambda. Dengan kebijakan ini, Amazon Cognito dapat menjalankan fungsi yang Anda pilih. Jika kredensial yang masuk tidak memiliki izin IAM yang memadai, Anda harus memperbarui kebijakan berbasis sumber daya secara terpisah. Untuk informasi selengkapnya, lihat [the section called "Pertimbangan penting"](#).

8. Pilih Save changes (Simpan perubahan).
9. Anda dapat menggunakan CloudWatch di konsol Lambda untuk mencatat fungsi Lambda Anda. Untuk informasi selengkapnya, lihat [Mengakses CloudWatch Log untuk Lambda](#).

## Acara pemicu Lambda kumpulan pengguna

Amazon Cognito meneruskan informasi peristiwa ke fungsi Lambda Anda. Fungsi Lambda mengembalikan objek peristiwa yang sama kembali ke Amazon Cognito dengan perubahan apa pun dalam respons. Peristiwa ini menunjukkan parameter umum pemicu Lambda:

### JSON

```
{
  "version": "string",
  "triggerSource": "string",
  "region": AWSRegion,
  "userPoolId": "string",
  "userName": "string",
  "callerContext":
    {
      "awsSdkVersion": "string",
      "clientId": "string"
    },
  "request":
    {
      "userAttributes": {
        "string": "string",
        ....
      }
    },
  "response": {}
}
```

## Kumpulan pengguna Lambda memicu parameter umum

### versi

Nomor versi fungsi Lambda Anda.

### triggerSource

Nama peristiwa yang memicu fungsi Lambda. Untuk deskripsi dari setiap triggerSource, lihat [Menghubungkan pemicu Lambda ke operasi fungsional kumpulan pengguna](#).

### wilayah

Wilayah AWS sebagai *AWSRegion* contoh.



## userPoolId

ID dari kumpulan pengguna.

## userName

Nama pengguna saat ini.

## callerContext

Metadata tentang permintaan dan lingkungan kode. Ini berisi bidang `awsSdkVersion` dan `clientId`.

### awsSdkVersion

Versi AWS SDK yang menghasilkan permintaan.

### ClientId

ID klien aplikasi kumpulan pengguna.

## request

Detail permintaan API pengguna Anda. Ini mencakup bidang berikut, dan parameter permintaan apa pun yang khusus untuk pemicu. Misalnya, peristiwa yang dikirimkan Amazon Cognito ke pemicu pra-otentikasi juga akan berisi parameter. `userNotFound` Anda dapat memproses nilai parameter ini untuk mengambil tindakan kustom saat pengguna Anda mencoba masuk dengan nama pengguna yang tidak terdaftar.

### UserAttributes

Satu atau lebih pasangan kunci-nilai dari nama dan nilai atribut pengguna, misalnya.

```
"email": "john@example.com"
```

## response

Parameter ini tidak berisi informasi apa pun dalam permintaan asli. Fungsi Lambda Anda harus mengembalikan seluruh acara ke Amazon Cognito, dan menambahkan parameter pengembalian apa pun ke file. `response` Untuk melihat parameter pengembalian apa yang dapat disertakan fungsi Anda, lihat dokumentasi untuk pemicu yang ingin Anda gunakan.

## Menghubungkan operasi API ke pemicu Lambda

Bagian berikut menjelaskan pemicu Lambda yang dipanggil Amazon Cognito dari aktivitas di kumpulan pengguna Anda.

Saat aplikasi Anda menandatangani pengguna melalui API kumpulan pengguna Amazon Cognito, UI yang dihosting, atau titik akhir kumpulan pengguna, Amazon Cognito akan memanggil fungsi Lambda Anda berdasarkan konteks sesi. Untuk informasi selengkapnya tentang API kumpulan pengguna Amazon Cognito dan titik akhir kumpulan pengguna, lihat [Menggunakan API kumpulan pengguna Amazon Cognito dan titik akhir kumpulan pengguna](#). Tabel di bagian berikut menjelaskan peristiwa yang menyebabkan Amazon Cognito memanggil fungsi, dan `triggerSource` string yang disertakan Amazon Cognito dalam permintaan.

## Topik

- [Pemicu Lambda di Amazon Cognito API](#)
- [Pemicu Lambda untuk pengguna lokal Amazon Cognito di UI yang dihosting](#)
- [Pemicu Lambda untuk pengguna federasi](#)

## Pemicu Lambda di Amazon Cognito API

Tabel berikut menjelaskan string sumber untuk pemicu Lambda yang dapat dipanggil Amazon Cognito saat aplikasi Anda membuat, masuk, atau memperbarui pengguna lokal.

### Sumber pemicu pengguna lokal di Amazon Cognito API

Operasi API	Pemicu Lambda	Sumber pemicu
<a href="#">AdminCreateUser</a>	Pra pendaftaran	PreSignUp_AdminCreateUser
	Pra generasi token	TokenGeneration_NewPasswordChallenge
	Pesan kustom	CustomMessage_AdminCreateUser
	Pengirim email kustom	CustomEmailSender_AdminCreateUser
	Pengirim SMS kustom	CustomSMSSender_AdminCreateUser
<a href="#">SignUp</a>	Pra pendaftaran	PreSignUp_SignUp

Operasi API	Pemicu Lambda	Sumber pemicu
	Pesan kustom	CustomMessage_SignUp
	Pengirim email kustom	CustomEmailSender_SignUp
	Pengirim SMS kustom	CustomSMSSender_SignUp
<a href="#">ConfirmSignUp</a> <a href="#">AdminConfirmSignUp</a>	Pasca konfirmasi	PostConfirmation_ConfirmSignUp
<a href="#">InitiateAuth</a> <a href="#">AdminInitiateAuth</a>	Pra autentikasi	PreAuthentication_Authentication
	Menentukan tantangan autentikasi	DefineAuthChallenge_Authentication
	Membuat tantangan autentikasi	CreateAuthChallenge_Authentication
	Pra generasi token	TokenGeneration_Authentication TokenGeneration_AuthenticateDevice TokenGeneration_RefreshTokens
	Migrasikan pengguna	UserMigration_Authentication
	Pesan kustom	CustomMessage_Authentication

Operasi API	Pemicu Lambda	Sumber pemicu
	Pengirim email kustom	CustomEmailSender_AccountTakeOverNotification
	Pengirim SMS kustom	CustomSMSSender_Authentication
<a href="#">ForgotPassword</a>	Migrasikan pengguna	UserMigration_ForgotPassword
	Pesan kustom	CustomMessage_ForgotPassword
	Pengirim email kustom	CustomEmailSender_ForgotPassword
	Pengirim SMS kustom	CustomSMSSender_ForgotPassword
<a href="#">ConfirmForgotPassword</a>	Pasca konfirmasi	PostConfirmation_ConfirmForgotPassword
<a href="#">UpdateUserAttributes</a>	Pesan kustom	CustomMessage_UpdateUserAttribute
<a href="#">AdminUpdateUserAttributes</a>	Pengirim email kustom	CustomEmailSender_UpdateUserAttribute
	Pengirim SMS kustom	CustomSMSSender_UpdateUserAttribute
<a href="#">VerifyUserAttributes</a>	Pesan kustom	CustomMessage_VerifyUserAttribute
	Pengirim email kustom	CustomEmailSender_VerifyUserAttribute

Operasi API	Pemicu Lambda	Sumber pemicu
	Pengirim SMS kustom	CustomSMSSender_VerifyUserAttribute

## Pemicu Lambda untuk pengguna lokal Amazon Cognito di UI yang dihosting

Tabel berikut menjelaskan string sumber untuk pemicu Lambda yang dapat dipanggil Amazon Cognito saat pengguna lokal masuk ke kumpulan pengguna Anda dengan UI yang dihosting.

Sumber pemicu pengguna lokal di UI yang dihosting

UI URI yang dihosting	Pemicu Lambda	Sumber pemicu
/signup	Pra pendaftaran	PreSignUp_SignUp
	Pesan kustom	CustomMessage_SignUp
	Pengirim email kustom	CustomEmailSender_SignUp
	Pengirim SMS kustom	CustomSMSSender_SignUp
/confirmuser	Pasca konfirmasi	PostConfirmation_ConfirmSignUp
/login	Pra autentikasi	PreAuthentication_Authentication
	Menentukan tantangan autentikasi	DefineAuthChallenge_Authentication
	Membuat tantangan autentikasi	CreateAuthChallenge_Authentication
	Pra generasi token	TokenGeneration_Authentication

UI URI yang dihosting	Pemicu Lambda	Sumber pemicu
		TokenGeneration_AuthenticateDevice
		TokenGeneration_RefreshTokens
	Migrasikan pengguna	UserMigration_Authentication
	Pesan kustom	CustomMessage_Authentication
	Pengirim email kustom	CustomEmailSender_AccountTakeOverNotification
	Pengirim SMS kustom	CustomSMSSender_Authentication
/forgotpassword	Migrasikan pengguna	UserMigration_ForgotPassword
	Pesan kustom	CustomMessage_ForgotPassword
	Pengirim email kustom	CustomEmailSender_ForgotPassword
	Pengirim SMS kustom	CustomSMSSender_ForgotPassword
/confirmforgotpassword	Pasca konfirmasi	PostConfirmation_ConfirmForgotPassword

## Pemicu Lambda untuk pengguna federasi

Anda dapat menggunakan pemicu Lambda berikut untuk menyesuaikan alur kerja kumpulan pengguna bagi pengguna yang masuk dengan penyedia federasi.

### Note

Pengguna gabungan dapat menggunakan UI yang dihosting Amazon Cognito untuk masuk, atau Anda dapat membuat permintaan ke [Otorisasi titik akhir](#) yang secara diam-diam mengalihkan mereka ke halaman login penyedia identitas mereka. Anda tidak dapat masuk ke pengguna gabungan dengan API kumpulan pengguna Amazon Cognito.

### Sumber pemicu pengguna federasi

Acara masuk	Pemicu Lambda	Sumber pemicu
Masuk pertama	Pra pendaftaran	PreSignUp_External Provider
	Pasca konfirmasi	PostConfirmation_ConfirmSignUp
	Pra generasi token	TokenGeneration_HostedAuth
Masuk selanjutnya	Pra autentikasi	PreAuthentication_Authentication
	Pasca autentikasi	PostAuthentication_Authentication
	Pra generasi token	TokenGeneration_HostedAuth

Masuk federasi tidak memanggil [Tantangan otentikasi kustom pemicu Lambda](#), [Memigrasi pengguna pemicu Lambda](#) [Pesan khusus Lambda pemicu](#), atau [Pemicu Lambda pengirim kustom](#) di kumpulan pengguna Anda.

## Menghubungkan pemacu Lambda ke operasi fungsional kumpulan pengguna

Setiap pemacu Lambda memiliki peran fungsional di kumpulan pengguna Anda. Misalnya, pemacu dapat mengubah alur pendaftaran Anda, atau menambahkan tantangan otentikasi khusus. Peristiwa yang dikirim Amazon Cognito ke fungsi Lambda dapat mencerminkan salah satu dari beberapa tindakan yang membentuk peran fungsional itu. Misalnya, Amazon Cognito memanggil pemacu pra-pendaftaran saat pengguna mendaftar, dan saat Anda membuat pengguna. Masing-masing kasus yang berbeda untuk peran fungsional yang sama memiliki `triggerSource` nilainya sendiri. Fungsi Lambda Anda dapat memproses peristiwa yang masuk secara berbeda berdasarkan operasi yang memanggilnya.

Amazon Cognito juga memanggil semua fungsi yang ditetapkan saat peristiwa sesuai dengan sumber pemacu. Misalnya, saat pengguna masuk ke kumpulan pengguna tempat Anda menetapkan pemacu migrasi dan pra autentikasi, mereka mengaktifkan keduanya.

Pemacu pendaftaran, konfirmasi, dan masuk (otentikasi)

Pemacu	Nilai <code>triggerSource</code>	Peristiwa
Pra pendaftaran	<code>PreSignUp_SignUp</code>	Pra pendaftaran.
Pra pendaftaran	<code>PreSignUp_AdminCreateUser</code>	Pra pendaftaran saat admin membuat pengguna baru.
Pra pendaftaran	<code>PreSignUp_ExternalProvider</code>	Pra pendaftaran untuk penyedia identitas eksternal.
Pasca konfirmasi	<code>PostConfirmation_ConfirmSignUp</code>	Pasca konfirmasi pendaftaran.
Pasca konfirmasi	<code>PostConfirmation_ConfirmForgotPassword</code>	Pasca konfirmasi Lupa Kata Sandi.
Pra autentikasi	<code>PreAuthentication_Authentication</code>	Pra autentikasi.



Pemicu	Nilai triggerSource	Peristiwa
Pasca autentikasi	PostAuthentication_Authentication	Pasca autentikasi.

#### Pemicu tantangan autentikasi kustom

Pemicu	Nilai triggerSource	Peristiwa
Menentukan tantangan autentikasi	DefineAuthChallenge_Authentication	Menentukan Tantangan Autentikasi.
Membuat tantangan autentikasi	CreateAuthChallenge_Authentication	Membuat Tantangan Autentikasi.
Memverifikasi tantangan autentikasi	VerifyAuthChallengeResponse_Authentication	Memverifikasi Respons Tantangan Autentikasi.

#### Pemicu pra generasi token

Pemicu	Nilai triggerSource	Peristiwa
Pra generasi token	TokenGeneration_HostedAuth	Amazon Cognito mengautentikasi pengguna dari halaman login UI yang dihosting.
Pra generasi token	TokenGeneration_Authentication	Alur otentikasi pengguna selesai.
Pra generasi token	TokenGeneration_NewPasswordChallenge	Admin menciptakan pengguna. Amazon Cognito memanggil ini ketika pengguna harus mengubah kata sandi sementara.
Pra generasi token	TokenGeneration_AuthenticateDevice	Akhir otentikasi perangkat pengguna.

Pemicu	Nilai triggerSource	Peristiwa
Pra generasi token	TokenGeneration_RefreshTokens	Pengguna mencoba menyegarkan identitas dan token akses.

### Memigrasi pemicu pengguna

Pemicu	Nilai triggerSource	Peristiwa
Migrasi pengguna	UserMigration_Authentication	Migrasi pengguna pada saat login.
Migrasi pengguna	UserMigration_ForgotPassword	Migrasi pengguna selama alur lupa kata sandi.

### Pesan kustom pemicu

Pemicu	Nilai triggerSource	Peristiwa
Pesan kustom	CustomMessage_SignUp	Pesan khusus saat pengguna mendaftar di kumpulan pengguna Anda.
Pesan kustom	CustomMessage_AdminCreateUser	Pesan khusus saat Anda membuat pengguna sebagai administrator dan Amazon Cognito mengirim mereka kata sandi sementara.
Pesan kustom	CustomMessage_ResendCode	Pesan kustom saat pengguna Anda yang ada meminta kode konfirmasi baru.
Pesan kustom	CustomMessage_ForgotPassword	Pesan kustom saat pengguna Anda meminta pengaturan ulang kata sandi.

Pemicu	Nilai triggerSource	Peristiwa
Pesan kustom	CustomMessage_UpdateUserAttribute	Pesan khusus saat pengguna mengubah alamat email atau nomor teleponnya dan Amazon Cognito mengirimkan kode verifikasi.
Pesan kustom	CustomMessage_VerifyUserAttribute	Pesan khusus saat pengguna menambahkan alamat email atau nomor telepon dan Amazon Cognito mengirimkan kode verifikasi.
Pesan kustom	CustomMessage_Authentication	Pesan khusus saat pengguna yang telah mengonfigurasi SMS MFA masuk.

## Pemicu Lambda pra pendaftaran

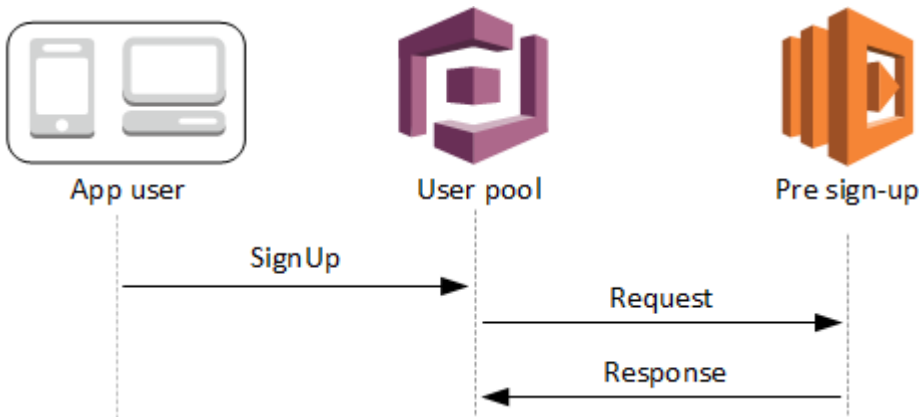
Sesaat sebelum Amazon Cognito mendaftarkan pengguna baru, ini mengaktifkan fungsi pra-pendaftaran. AWS Lambda Sebagai bagian dari proses pendaftaran, Anda dapat menggunakan fungsi ini untuk melakukan validasi khusus dan, berdasarkan hasil validasi Anda, menerima atau menolak permintaan pendaftaran.

### Topik

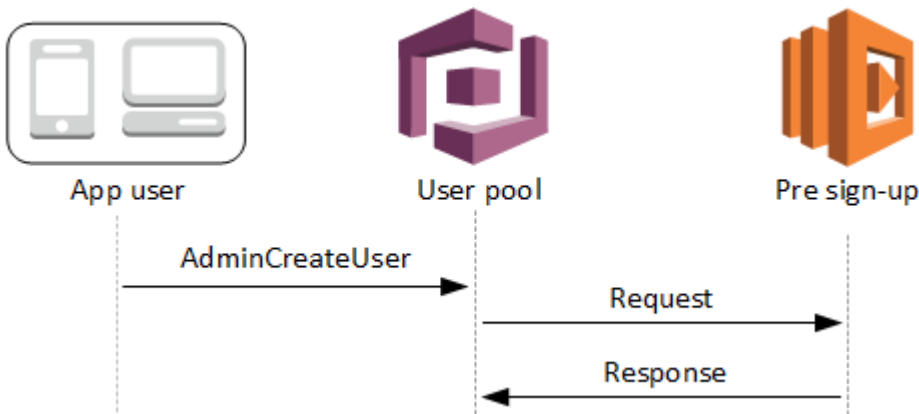
- [Aliran Lambda pra pendaftaran](#)
- [Parameter pemicu Lambda pra-pendaftaran](#)
- [Tutorial pendaftaran](#)
- [Contoh pra pendaftaran: Konfirmasi otomatis pengguna dari domain terdaftar](#)
- [Contoh pra pendaftaran: Konfirmasi otomatis dan verifikasi otomatis semua pengguna](#)
- [Contoh pra pendaftaran: Tolak pendaftaran jika nama pengguna memiliki kurang dari lima karakter](#)

## Aliran Lambda pra pendaftaran

### Alur pendaftaran klien



### Alur pendaftaran server



Permintaan tersebut mencakup data validasi dari klien. Data ini berasal dari `ValidationData` nilai yang diteruskan ke kumpulan pengguna `SignUp` dan metode `AdminCreateUser` API.

### Parameter pemacu Lambda pra-pendaftaran

Permintaan yang diteruskan Amazon Cognito ke fungsi Lambda ini adalah kombinasi dari parameter di bawah ini dan parameter [umum yang ditambahkan Amazon](#) Cognito ke semua permintaan.

### JSON

```

{
  "request": {
    "userAttributes": {
      "string": "string",
      . . .
    }
  }
}
  
```

```
    },
    "validationData": {
      "string": "string",
      . . .
    },
    "clientMetadata": {
      "string": "string",
      . . .
    }
  },

  "response": {
    "autoConfirmUser": "boolean",
    "autoVerifyPhone": "boolean",
    "autoVerifyEmail": "boolean"
  }
}
```

## Parameter permintaan pra-pendaftaran

### userAttributes

Satu atau lebih pasangan nilai-nama yang mewakili atribut pengguna. Nama atribut adalah kuncinya.

### validationData

Satu atau beberapa pasangan nilai kunci dengan data atribut pengguna yang diteruskan aplikasi Anda ke Amazon Cognito dalam permintaan untuk membuat pengguna baru. Kirim informasi ini ke fungsi Lambda Anda dalam ValidationData parameter permintaan Anda [AdminCreateUser](#) atau [SignUpAPI](#).

Amazon Cognito tidak menyetel ValidationData data Anda sebagai atribut pengguna yang Anda buat. ValidationData adalah informasi pengguna sementara yang Anda berikan untuk tujuan pemicu Lambda pra pendaftaran Anda.

### clientMetadata

Satu atau lebih pasangan nilai-kunci yang dapat Anda berikan sebagai masukan kustom ke fungsi Lambda yang Anda tentukan untuk pemicu pra pendaftaran. Anda dapat meneruskan data ini ke fungsi Lambda dengan menggunakan ClientMetadata parameter dalam tindakan API berikut: [AdminCreateUser](#), [AdminRespondToAuthChallengeForgotPassword](#), dan [SignUp](#)

## Parameter respons pra pendaftaran

Dalam respons, Anda dapat mengatur `autoConfirmUser` ke `true` jika Anda ingin mengonfirmasi pengguna secara otomatis. Anda dapat mengatur `autoVerifyEmail` ke `true` untuk memverifikasi secara otomatis email pengguna. Anda dapat mengatur `autoVerifyPhone` ke `true` untuk memverifikasi secara otomatis nomor telepon pengguna.

### Note

Parameter respons `autoVerifyPhone`, `autoVerifyEmail` dan `autoConfirmUser` diabaikan oleh Amazon Cognito saat fungsi Lambda pra pendaftaran dipicu oleh API `AdminCreateUser`.

### `autoConfirmUser`

Atur ke `true` untuk mengonfirmasi otomatis pengguna, atau `false` untuk sebaliknya.

### `autoVerifyEmail`

Setel `true` untuk menetapkan alamat email pengguna yang mendaftar sebagai terverifikasi, atau `false` sebaliknya. Jika `autoVerifyEmail` diatur ke `true`, atribut `email` harus memiliki nilai yang valid dan bukan nol. Jika tidak, kesalahan akan terjadi dan pengguna tidak akan bisa menyelesaikan pendaftaran.

Jika `email` atribut dipilih sebagai alias, alias akan dibuat untuk alamat email pengguna saat `autoVerifyEmail` disetel. Jika alias dengan alamat email tersebut sudah ada, alias akan dipindahkan ke pengguna baru dan alamat email pengguna sebelumnya akan ditandai sebagai tidak diverifikasi. Untuk informasi selengkapnya, lihat [Menyesuaikan atribut masuk](#).

### `autoVerifyPhone`

Atur ke `true` untuk mengatur sebagai terverifikasi nomor telepon dari pengguna yang mendaftar, atau `false` untuk sebaliknya. Jika `autoVerifyPhone` diatur ke `true`, atribut `phone_number` harus memiliki nilai yang valid dan bukan nol. Jika tidak, kesalahan akan terjadi dan pengguna tidak akan bisa menyelesaikan pendaftaran.

Jika atribut `phone_number` dipilih sebagai alias, alias akan dibuat untuk nomor telepon pengguna ketika `autoVerifyPhone` diatur. Jika alias dengan nomor telepon tersebut sudah ada, alias akan dipindahkan ke pengguna baru dan nomor telepon pengguna sebelumnya akan ditandai sebagai tidak terverifikasi. Untuk informasi lebih lanjut, lihat [Menyesuaikan atribut masuk](#).

## Tutorial pendaftaran

Fungsi Lambda pra pendaftaran dipicu sebelum pengguna mendaftar. Lihat tutorial pendaftaran Amazon Cognito ini untuk JavaScript, Android, dan iOS.

Platform	Tutorial
JavaScript SDK Identitas	<a href="#">Mendaftar pengguna dengan JavaScript</a>
SDK Identitas Android	<a href="#">Mendaftar pengguna dengan Android</a>
SDK Identitas iOS	<a href="#">Mendaftar pengguna dengan iOS</a>

### Contoh pra pendaftaran: Konfirmasi otomatis pengguna dari domain terdaftar

Anda dapat menggunakan pemacu Lambda pra pendaftaran untuk menambahkan logika khusus yang memvalidasi pengguna baru yang mendaftar ke kumpulan pengguna Anda. Ini adalah contoh JavaScript program yang menunjukkan cara mendaftar pengguna baru. Ini memanggil pemacu Lambda pra-pendaftaran sebagai bagian dari otentikasi.

#### JavaScript

```
var attributeList = [];  
var dataEmail = {  
  Name: "email",  
  Value: "...", // your email here  
};  
var dataPhoneNumber = {  
  Name: "phone_number",  
  Value: "...", // your phone number here with +country code and no delimiters in  
  front  
};  
  
var dataEmailDomain = {  
  Name: "custom:domain",  
  Value: "example.com",  
};  
var attributeEmail = new AmazonCognitoIdentity.CognitoUserAttribute(dataEmail);  
var attributePhoneNumber = new AmazonCognitoIdentity.CognitoUserAttribute(  
  dataPhoneNumber  
);
```

```
var attributeEmailDomain = new AmazonCognitoIdentity.CognitoUserAttribute(  
    dataEmailDomain  
);  
  
attributeList.push(attributeEmail);  
attributeList.push(attributePhoneNumber);  
attributeList.push(attributeEmailDomain);  
  
var cognitoUser;  
userPool.signUp(  
    "username",  
    "password",  
    attributeList,  
    null,  
    function (err, result) {  
        if (err) {  
            alert(err);  
            return;  
        }  
        cognitoUser = result.user;  
        console.log("user name is " + cognitoUser.getUsername());  
    }  
);
```

Ini adalah sampel pemicu Lambda yang dipanggil tepat sebelum mendaftar dengan pemicu Lambda pra pendaftaran kolam pengguna. Ini menggunakan atribut kustom custom:domain untuk mengonfirmasi pengguna baru secara otomatis dari domain email tertentu. Setiap pengguna baru yang tidak berada dalam domain kustom akan ditambahkan ke kolam pengguna, tetapi tidak dikonfirmasi secara otomatis.

## Node.js

```
exports.handler = (event, context, callback) => {  
    // Set the user pool autoConfirmUser flag after validating the email domain  
    event.response.autoConfirmUser = false;  
  
    // Split the email address so we can compare domains  
    var address = event.request.userAttributes.email.split("@");  
  
    // This example uses a custom attribute "custom:domain"  
    if (event.request.userAttributes.hasOwnProperty("custom:domain")) {  
        if (event.request.userAttributes["custom:domain"] === address[1]) {
```



```
        event.response.autoConfirmUser = true;
    }
}

// Return to Amazon Cognito
callback(null, event);
};
```

## Python

```
def lambda_handler(event, context):
    # It sets the user pool autoConfirmUser flag after validating the email domain
    event['response']['autoConfirmUser'] = False

    # Split the email address so we can compare domains
    address = event['request']['userAttributes']['email'].split('@')

    # This example uses a custom attribute 'custom:domain'
    if 'custom:domain' in event['request']['userAttributes']:
        if event['request']['userAttributes']['custom:domain'] == address[1]:
            event['response']['autoConfirmUser'] = True

    # Return to Amazon Cognito
    return event
```

Amazon Cognito meneruskan informasi peristiwa ke fungsi Lambda Anda. Fungsi kemudian mengembalikan objek acara yang sama ke Amazon Cognito, dengan perubahan apa pun dalam respons. Di konsol Lambda, Anda dapat mengatur peristiwa pengujian dengan data yang relevan dengan pemicu Lambda Anda. Berikut ini adalah peristiwa pengujian untuk sampel kode ini:

## JSON

```
{
  "request": {
    "userAttributes": {
      "email": "testuser@example.com",
      "custom:domain": "example.com"
    }
  },
  "response": {}
}
```

## Contoh pra pendaftaran: Konfirmasi otomatis dan verifikasi otomatis semua pengguna

Contoh ini mengonfirmasi semua pengguna dan mengatur atribut `email` dan `phone_number` pengguna menjadi terverifikasi jika atribut tersebut ada. Juga, jika aliasing diaktifkan, alias akan dibuat untuk `phone_number` dan `email` saat verifikasi otomatis diatur.

### Note

Jika alias dengan nomor telepon yang sama sudah ada, alias akan dipindahkan ke pengguna baru, dan `phone_number` pengguna sebelumnya akan ditandai sebagai belum diverifikasi. Hal yang sama berlaku untuk alamat email. Untuk mencegah hal ini terjadi, Anda dapat menggunakan [ListUsers API](#) kumpulan pengguna untuk melihat apakah ada pengguna yang sudah menggunakan nomor telepon atau alamat email pengguna baru sebagai alias.

### Node.js

```
const handler = async (event) => {
  // Confirm the user
  event.response.autoConfirmUser = true;
  // Set the email as verified if it is in the request
  if (event.request.userAttributes.hasOwnProperty("email")) {
    event.response.autoVerifyEmail = true;
  }

  // Set the phone number as verified if it is in the request
  if (event.request.userAttributes.hasOwnProperty("phone_number")) {
    event.response.autoVerifyPhone = true;
  }

  return event;
};

export { handler };
```

### Python

```
def lambda_handler(event, context):
    # Confirm the user
    event['response']['autoConfirmUser'] = True
```

```
# Set the email as verified if it is in the request
if 'email' in event['request']['userAttributes']:
    event['response']['autoVerifyEmail'] = True

# Set the phone number as verified if it is in the request
if 'phone_number' in event['request']['userAttributes']:
    event['response']['autoVerifyPhone'] = True

# Return to Amazon Cognito
return event
```

Amazon Cognito meneruskan informasi peristiwa ke fungsi Lambda Anda. Fungsi kemudian mengembalikan objek acara yang sama ke Amazon Cognito, dengan perubahan apa pun dalam respons. Di konsol Lambda, Anda dapat mengatur peristiwa pengujian dengan data yang relevan dengan pemicu Lambda Anda. Berikut ini adalah peristiwa pengujian untuk sampel kode ini:

## JSON

```
{
  "request": {
    "userAttributes": {
      "email": "user@example.com",
      "phone_number": "+12065550100"
    }
  },
  "response": {}
}
```

Contoh pra pendaftaran: Tolak pendaftaran jika nama pengguna memiliki kurang dari lima karakter

Contoh ini memeriksa panjang nama pengguna dalam permintaan pendaftaran. Contoh mengembalikan kesalahan jika pengguna telah memasukkan nama yang panjangnya kurang dari lima karakter.

## Node.js

```
exports.handler = (event, context, callback) => {
  // Impose a condition that the minimum length of the username is 5 is imposed on
  all user pools.
```

```
    if (event.userName.length < 5) {
        var error = new Error("Cannot register users with username less than the
minimum length of 5");
        // Return error to Amazon Cognito
        callback(error, event);
    }
    // Return to Amazon Cognito
    callback(null, event);
};
```

## Python

```
def lambda_handler(event, context):
    if len(event['userName']) < 5:
        raise Exception("Cannot register users with username less than the minimum
length of 5")
    # Return to Amazon Cognito
    return event
```

Amazon Cognito meneruskan informasi peristiwa ke fungsi Lambda Anda. Fungsi kemudian mengembalikan objek acara yang sama ke Amazon Cognito, dengan perubahan apa pun dalam respons. Di konsol Lambda, Anda dapat mengatur peristiwa pengujian dengan data yang relevan dengan pemicu Lambda Anda. Berikut ini adalah peristiwa pengujian untuk sampel kode ini:

## JSON

```
{
  "userName": "rroe",
  "response": {}
}
```

## Posting konfirmasi pemicu Lambda

Amazon Cognito memanggil pemicu ini setelah pengguna yang mendaftar mengonfirmasi akun pengguna mereka. Dalam fungsi Lambda konfirmasi posting Anda, Anda dapat mengirim pesan khusus atau menambahkan permintaan API kustom. Misalnya, Anda dapat menanyakan sistem eksternal dan mengisi atribut tambahan kepada pengguna. Amazon Cognito memanggil pemicu ini hanya untuk pengguna yang mendaftar di kumpulan pengguna Anda, bukan untuk akun pengguna yang Anda buat dengan kredensi administrator Anda.

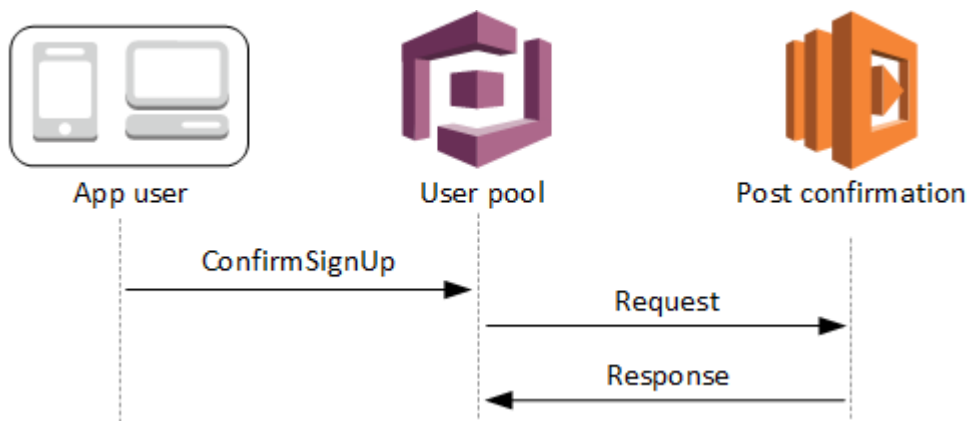
Permintaan berisi atribut saat ini untuk pengguna yang dikonfirmasi.

Topik

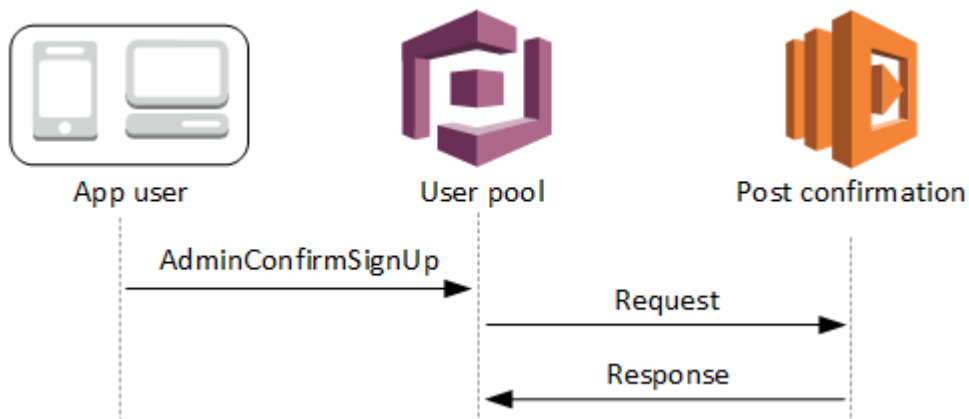
- [Posting konfirmasi aliran Lambda](#)
- [Posting konfirmasi parameter pemicu Lambda](#)
- [Tutorial konfirmasi pengguna](#)
- [Contoh konfirmasi posting](#)

## Posting konfirmasi aliran Lambda

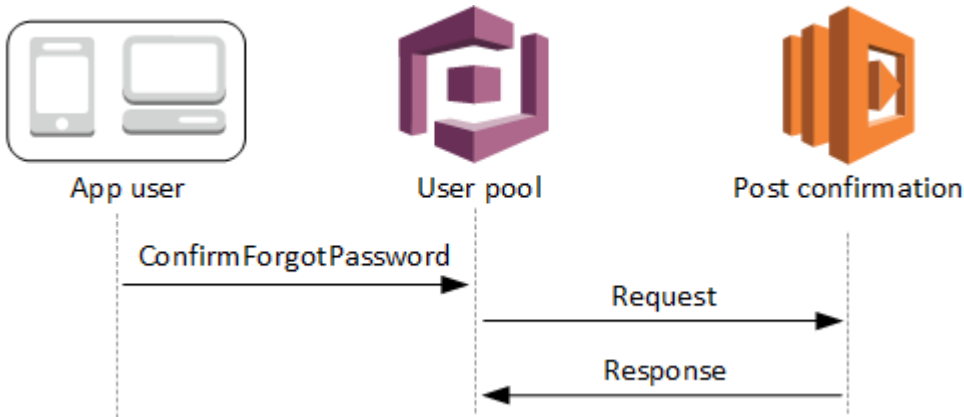
Alur klien mengonfirmasi pendaftaran



Server mengonfirmasi alur pendaftaran



## Alur konfirmasi lupa kata sandi



## Posting konfirmasi parameter pemicu Lambda

Permintaan yang diteruskan Amazon Cognito ke fungsi Lambda ini adalah kombinasi dari parameter di bawah ini dan [parameter umum](#) yang ditambahkan Amazon Cognito ke semua permintaan.

### JSON

```

{
  "request": {
    "userAttributes": {
      "string": "string",
      . . .
    },
    "clientMetadata": {
      "string": "string",
      . . .
    }
  },
  "response": {}
}
  
```

### Parameter permintaan konfirmasi posting

#### userAttributes

Satu atau lebih pasangan nilai-kunci yang mewakili atribut pengguna.

## clientMetadata

Satu atau lebih pasangan nilai-kunci yang dapat Anda berikan sebagai masukan kustom ke fungsi Lambda yang Anda tentukan untuk pemacu pasca konfirmasi. Anda dapat meneruskan data ini ke fungsi Lambda Anda dengan menggunakan `ClientMetadataParameter` dalam tindakan API berikut: [AdminConfirmSignUp](#), [ConfirmForgotPassword](#), [ConfirmSignUp](#), dan [SignUp](#).

## Parameter respons konfirmasi posting

Tidak ada informasi pengembalian tambahan yang diharapkan dalam respons.

## Tutorial konfirmasi pengguna

Fungsi Lambda pasca konfirmasi dipicu tepat setelah Amazon Cognito mengonfirmasi pengguna baru. Lihat tutorial konfirmasi pengguna ini untuk [JavaScript](#), [Android](#), dan [iOS](#).

Platform	Tutorial
JavaScript SDK Identitas	<a href="#">Konfirmasikan pengguna dengan JavaScript</a>
SDK Identitas Android	<a href="#">Konfirmasikan pengguna dengan Android</a>
SDK Identitas iOS	<a href="#">Konfirmasikan pengguna dengan iOS</a>

## Contoh konfirmasi posting

Contoh fungsi Lambda ini mengirimkan pesan email konfirmasi kepada pengguna Anda menggunakan Amazon SES. Untuk informasi selengkapnya lihat [Panduan Developer Amazon Simple Email Service](#).

### Node.js

```
// Import required AWS SDK clients and commands for Node.js. Note that this requires
// the `@aws-sdk/client-ses` module to be either bundled with this code or included
// as a Lambda layer.
import { SES, SendEmailCommand } from "@aws-sdk/client-ses";
const ses = new SES();

const handler = async (event) => {
  if (event.request.userAttributes.email) {
```

```
    await sendTheEmail(
      event.request.userAttributes.email,
      `Congratulations ${event.userName}, you have been confirmed.`
    );
  }
  return event;
};

const sendTheEmail = async (to, body) => {
  const eParams = {
    Destination: {
      ToAddresses: [to],
    },
    Message: {
      Body: {
        Text: {
          Data: body,
        },
      },
      Subject: {
        Data: "Cognito Identity Provider registration completed",
      },
    },
    // Replace source_email with your SES validated email address
    Source: "<source_email>",
  };
  try {
    await ses.send(new SendEmailCommand(eParams));
  } catch (err) {
    console.log(err);
  }
};

export { handler };
```

Amazon Cognito meneruskan informasi peristiwa ke fungsi Lambda Anda. Fungsi kemudian mengembalikan objek acara yang sama ke Amazon Cognito, dengan perubahan apa pun dalam respons. Di konsol Lambda, Anda dapat mengatur peristiwa pengujian dengan data yang relevan dengan pemicu Lambda Anda. Berikut ini adalah peristiwa pengujian untuk sampel kode ini:



## JSON

```
{
  "request": {
    "userAttributes": {
      "email": "user@example.com",
      "email_verified": true
    }
  },
  "response": {}
}
```

## Pemicu Lambda pra otentikasi

Amazon Cognito memanggil pemicu ini saat pengguna mencoba masuk sehingga Anda dapat membuat validasi khusus yang melakukan tindakan persiapan. Misalnya, Anda dapat menolak permintaan otentikasi atau merekam data sesi ke sistem eksternal.

### Note

Pemicu Lambda ini tidak aktif saat pengguna tidak ada, atau sudah memiliki sesi yang ada di kumpulan pengguna Anda. Jika `PreventUserExistenceErrors` pengaturan klien aplikasi kumpulan pengguna disetel ke `ENABLED`, maka pemicu Lambda akan diaktifkan.

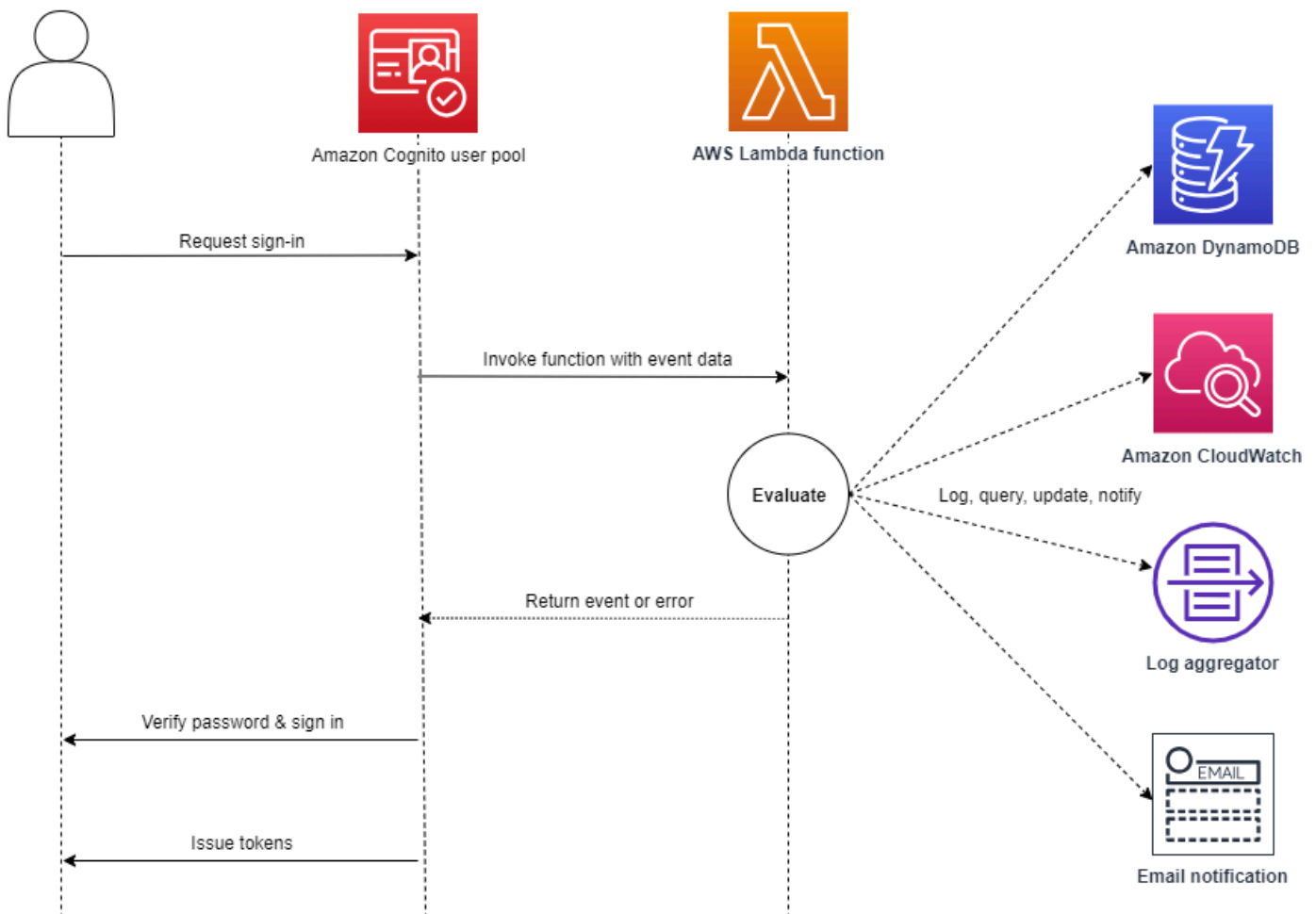
### Topik

- [Ikhtisar aliran otentikasi](#)
- [Parameter pemicu Lambda pra otentikasi](#)
- [Contoh pra otentikasi](#)

## Ikhtisar aliran otentikasi

## Amazon Cognito pre authentication trigger

Evaluate and authorize user sign-in



Permintaan tersebut menyertakan data validasi klien dari `ClientMetadata` nilai yang diteruskan aplikasi Anda ke kumpulan pengguna `InitiateAuth` dan operasi `AdminInitiateAuth` API.

Untuk informasi selengkapnya, lihat [Alur otentikasi kumpulan pengguna](#).

### Parameter pemicu Lambda pra otentikasi

Permintaan yang diteruskan Amazon Cognito ke fungsi Lambda ini adalah kombinasi dari parameter di bawah ini dan parameter [umum yang ditambahkan Amazon Cognito](#) ke semua permintaan.

## JSON

```
{
  "request": {
    "userAttributes": {
      "string": "string",
      . . .
    },
    "validationData": {
      "string": "string",
      . . .
    },
    "userNotFound": boolean
  },
  "response": {}
}
```

### Parameter permintaan pra otentikasi

#### userAttributes

Satu atau lebih pasangan nama-nilai yang mewakili atribut pengguna.

#### userNotFound

Saat Anda mengatur `PreventUserExistenceErrors` `ENABLED` untuk klien kumpulan pengguna Anda, Amazon Cognito mengisi Boolean ini.

#### validationData

Satu atau beberapa pasangan nilai kunci yang berisi data validasi dalam permintaan login pengguna. Untuk meneruskan data ini ke fungsi Lambda Anda, gunakan `ClientMetadata` parameter dalam tindakan [InitiateAuth](#) dan [AdminInitiateAuth](#) API.

### Parameter respons pra otentikasi

Amazon Cognito tidak mengharapkan informasi pengembalian tambahan dalam tanggapan. Fungsi Anda dapat menampilkan kesalahan untuk menolak upaya masuk, atau menggunakan operasi API untuk menanyakan dan memodifikasi sumber daya Anda.

## Contoh pra otentikasi

Fungsi contoh ini mencegah pengguna masuk ke kumpulan pengguna Anda dengan klien aplikasi tertentu. Karena fungsi Lambda pra autentikasi tidak dipanggil saat pengguna memiliki sesi yang sudah ada, fungsi ini hanya mencegah sesi baru dengan ID klien aplikasi yang ingin Anda blokir.

### Node.js

```
const handler = async (event) => {
  if (
    event.callerContext.clientId === "user-pool-app-client-id-to-be-blocked"
  ) {
    throw new Error("Cannot authenticate users from this user pool app client");
  }

  return event;
};

export { handler };
```

### Python

```
def lambda_handler(event, context):
    if event['callerContext']['clientId'] == "<user pool app client id to be
    blocked>":
        raise Exception("Cannot authenticate users from this user pool app client")

    # Return to Amazon Cognito
    return event
```

Amazon Cognito meneruskan informasi peristiwa ke fungsi Lambda Anda. Fungsi kemudian mengembalikan objek acara yang sama ke Amazon Cognito, dengan perubahan apa pun dalam respons. Di konsol Lambda, Anda dapat mengatur peristiwa pengujian dengan data yang relevan dengan pemicu Lambda Anda. Berikut ini adalah peristiwa pengujian untuk sampel kode ini:

### JSON

```
{
  "callerContext": {
    "clientId": "<user pool app client id to be blocked>"
  }
}
```

```
    },  
    "response": {}  
  }  
}
```

## Posting otentikasi pemicu Lambda

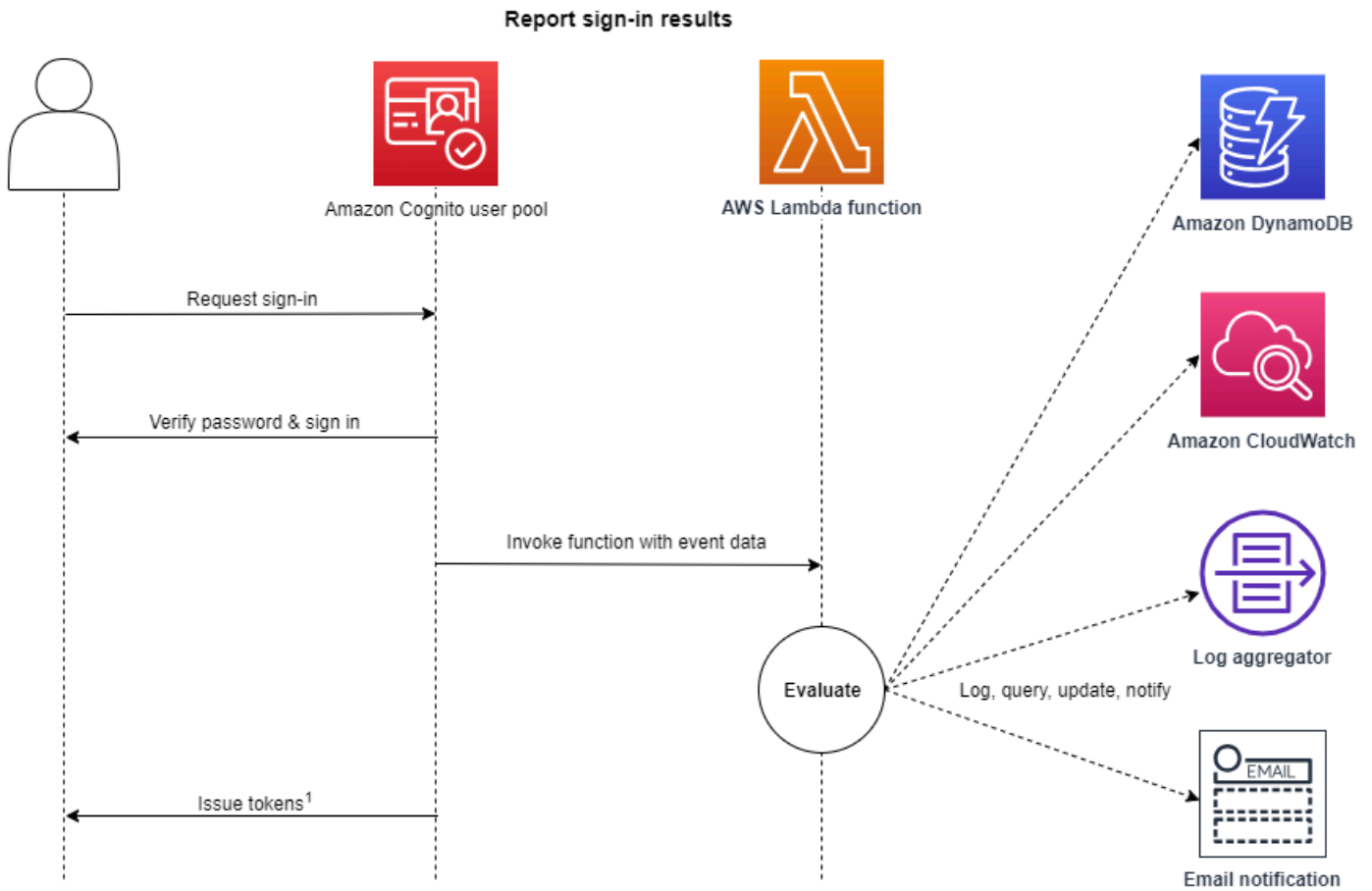
Karena Amazon Cognito memanggil pemicu ini setelah masuk ke pengguna, Anda dapat menambahkan logika khusus setelah Amazon Cognito mengautentikasi pengguna.

### Topik

- [Ikhtisar aliran otentikasi](#)
- [Posting otentikasi parameter pemicu Lambda](#)
- [Tutorial otentikasi](#)
- [Contoh otentikasi posting](#)

## Ikhtisar aliran otentikasi

### Amazon Cognito post authentication trigger



Untuk informasi selengkapnya, lihat [Alur otentikasi kumpulan pengguna](#).

### Posting otentikasi parameter pemicu Lambda

Permintaan yang diteruskan Amazon Cognito ke fungsi Lambda ini adalah kombinasi dari parameter di bawah ini dan parameter [umum yang ditambahkan Amazon](#) Cognito ke semua permintaan.

#### JSON

```

{
  "request": {
    "userAttributes": {
      "string": "string",
    }
  }
}
    
```

```
    . . .
  },
  "newDeviceUsed": boolean,
  "clientMetadata": {
    "string": "string",
    . . .
  }
},
"response": {}
}
```

## Parameter permintaan otentikasi posting

### newDeviceUsed

Bendera ini menunjukkan jika pengguna telah masuk pada perangkat baru. Amazon Cognito hanya menetapkan tanda ini jika nilai perangkat yang diingat dari kumpulan pengguna adalah Always atau User Opt-In

### userAttributes

Satu atau lebih pasangan nilai-nama yang mewakili atribut pengguna.

### clientMetadata

Satu atau lebih pasangan nilai-kunci yang dapat Anda berikan sebagai masukan kustom ke fungsi Lambda yang Anda tentukan untuk pemicu pasca autentikasi. Untuk meneruskan data ini ke fungsi Lambda Anda, Anda dapat menggunakan ClientMetadata parameter dalam tindakan [AdminRespondToAuthChallenge](#) dan [RespondToAuthChallenge](#) API. Amazon Cognito tidak menyertakan data dari ClientMetadata parameter dalam [AdminInitiateAuth](#) dan operasi [InitiateAuth](#) API dalam permintaan yang diteruskan ke fungsi otentikasi pos.

## Parameter respons otentikasi posting

Amazon Cognito tidak mengharapkan informasi pengembalian tambahan dalam tanggapan. Fungsi Anda dapat menggunakan operasi API untuk menanyakan dan memodifikasi sumber daya Anda, atau merekam metadata peristiwa ke sistem eksternal.

## Tutorial otentikasi

Segera setelah Amazon Cognito masuk ke pengguna, itu mengaktifkan fungsi Lambda otentikasi posting. Lihat tutorial masuk ini untuk JavaScript, Android, dan iOS.

Platform	Tutorial
JavaScript SDK Identitas	<a href="#">Masuk pengguna dengan JavaScript</a>
SDK Identitas Android	<a href="#">Masuk pengguna dengan Android</a>
SDK Identitas iOS	<a href="#">Masuk pengguna dengan iOS</a>

## Contoh otentikasi posting

Contoh otentikasi posting ini fungsi Lambda mengirimkan data dari login CloudWatch yang berhasil ke Log.

### Node.js

```
const handler = async (event) => {
  // Send post authentication data to Amazon CloudWatch logs
  console.log("Authentication successful");
  console.log("Trigger function =", event.triggerSource);
  console.log("User pool = ", event.userPoolId);
  console.log("App client ID = ", event.callerContext.clientId);
  console.log("User ID = ", event.userName);

  return event;
};

export { handler }
```

### Python

```
import os
def lambda_handler(event, context):

  # Send post authentication data to Cloudwatch logs
  print ("Authentication successful")
  print ("Trigger function =", event['triggerSource'])
  print ("User pool = ", event['userPoolId'])
  print ("App client ID = ", event['callerContext']['clientId'])
  print ("User ID = ", event['userName'])
```



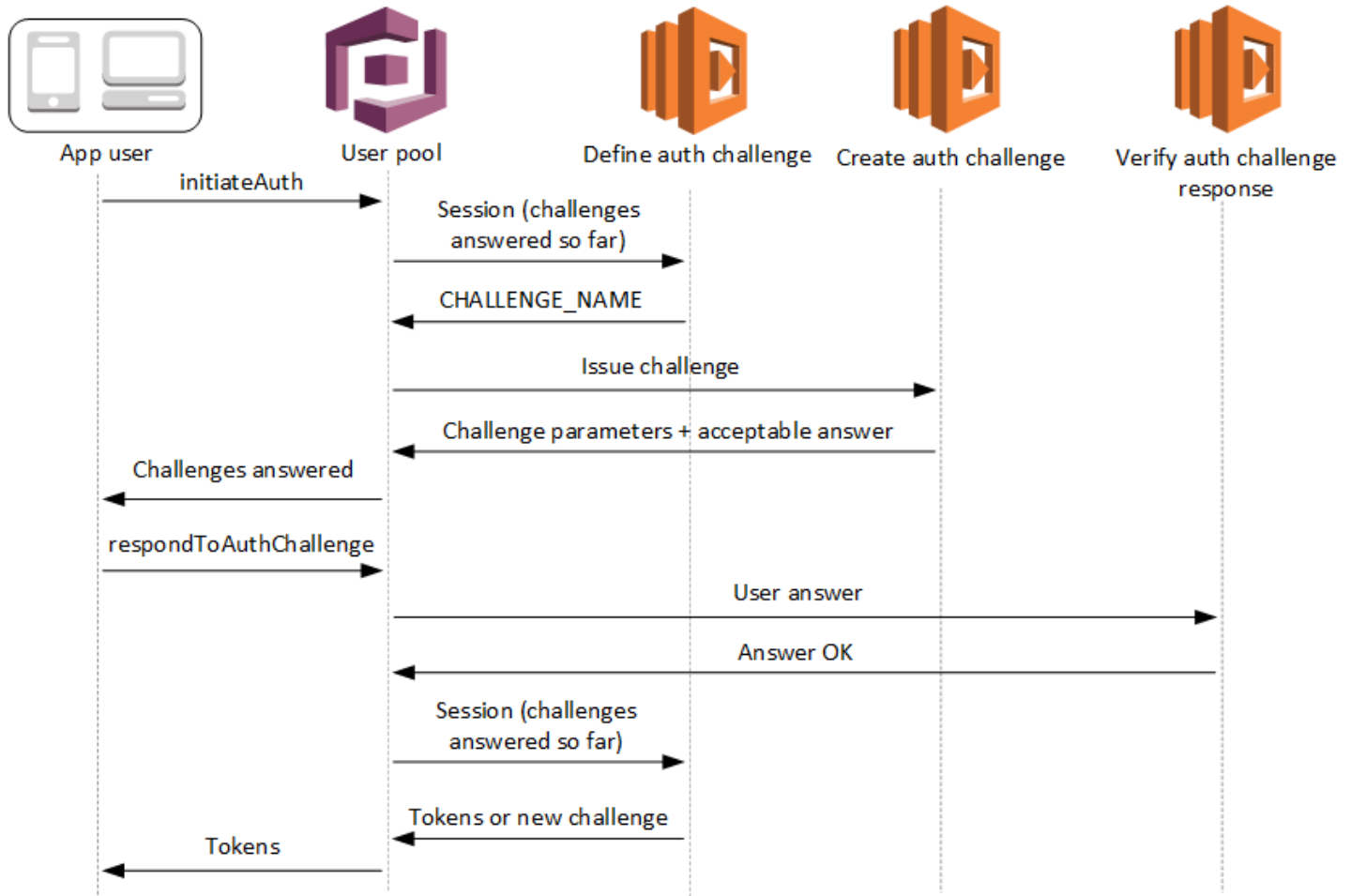
```
# Return to Amazon Cognito
return event
```

Amazon Cognito meneruskan informasi peristiwa ke fungsi Lambda Anda. Fungsi kemudian mengembalikan objek acara yang sama ke Amazon Cognito, dengan perubahan apa pun dalam respons. Di konsol Lambda, Anda dapat mengatur peristiwa pengujian dengan data yang relevan dengan pemicu Lambda Anda. Berikut ini adalah peristiwa pengujian untuk sampel kode ini:

## JSON

```
{
  "triggerSource": "testTrigger",
  "userPoolId": "testPool",
  "userName": "testName",
  "callerContext": {
    "clientId": "12345"
  },
  "response": {}
}
```

## Tantangan otentikasi kustom pemicu Lambda



Lambda ini memicu masalah dan memverifikasi tantangan mereka sendiri sebagai bagian dari [alur autentikasi kustom](#) kolam pengguna.

### Menentukan tantangan autentikasi

Amazon Cognito memanggil pemicu ini untuk memulai alur autentikasi kustom.

### Membuat tantangan autentikasi

Amazon Cognito memanggil pemicu ini setelah Menentukan Tantangan Autentikasi untuk membuat tantangan kustom.


### Memverifikasi respons tantangan autentikasi

Amazon Cognito memanggil pemicu ini untuk memverifikasi apakah respons dari pengguna akhir untuk tantangan kustom valid atau tidak.

Anda dapat menggabungkan jenis tantangan baru dengan pemicu Lambda tantangan ini. Misalnya, jenis tantangan ini mungkin menyertakan CAPTCHA atau pertanyaan tantangan dinamis.

Anda dapat menggeneralisasi otentikasi menjadi dua langkah umum dengan kumpulan pengguna `InitiateAuth` dan metode `RespondToAuthChallenge` API.

Dalam alur ini, pengguna mengautentikasi dengan menjawab tantangan berturut-turut sampai autentikasi gagal atau pengguna mengeluarkan token. Kedua panggilan API ini dapat diulang untuk menyertakan tantangan yang berbeda.

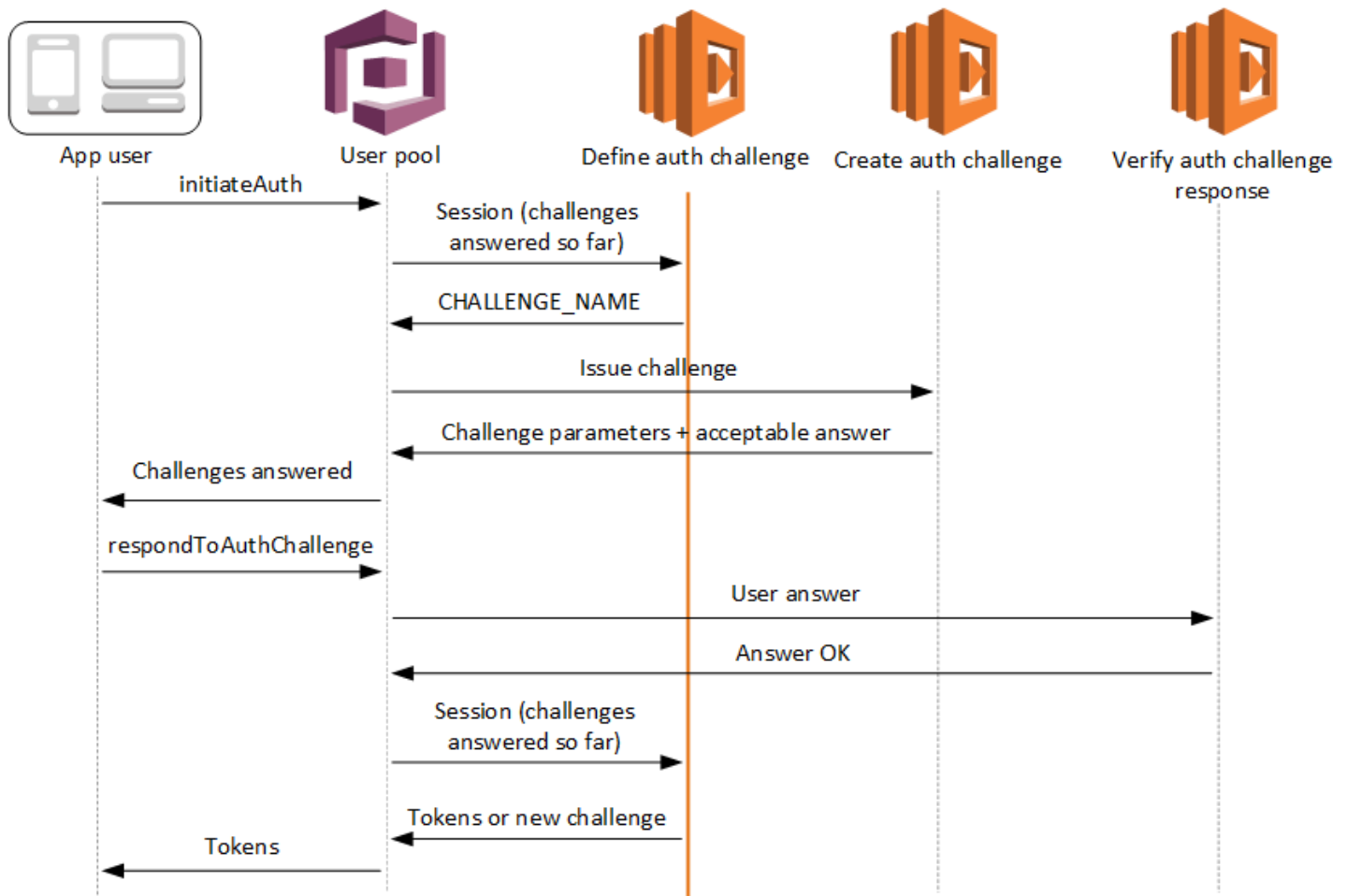
 Note

UI yang dihosting Amazon Cognito tidak mendukung otentikasi khusus dengan [tantangan otentikasi khusus](#) yang memicu Lambda.

## Topik

- [Tentukan tantangan Auth pemicu Lambda](#)
- [Membuat pemicu Lambda tantangan autentikasi](#)
- [Memverifikasi pemicu Lambda respon tantangan autentikasi](#)

## Tentukan tantangan Auth pemicu Lambda



### Menentukan tantangan autentikasi

Amazon Cognito memanggil pemicu ini untuk memulai [alur autentikasi kustom](#).

Permintaan untuk pemicu Lambda ini berisi `session sessionParameter` adalah array yang berisi semua tantangan yang disajikan kepada pengguna dalam proses otentikasi saat ini. Permintaan juga mencakup hasil yang sesuai. `sessionArray` menyimpan detail tantangan (`ChallengeResult`) dalam urutan kronologis. Tantangan tersebut `session[0]` merupakan tantangan pertama yang diterima pengguna.

Anda dapat meminta Amazon Cognito memverifikasi kata sandi pengguna sebelum mengeluarkan tantangan kustom Anda. Setiap pemicu Lambda yang terkait dalam kategori Otentikasi [kuota tingkat permintaan](#) akan berjalan saat Anda melakukan otentikasi SRP dalam alur tantangan khusus. Berikut adalah gambaran umum prosesnya:

1. Aplikasi Anda memulai proses masuk dengan memanggil `InitiateAuth` atau menggunakan `AdminInitiateAuth` peta. `AuthParameters` Parameter harus mencakup `CHALLENGE_NAME: SRP_A`, dan nilai untuk `SRP_A` dan `USERNAME`.
2. Amazon Cognito memanggil pemicu Lambda tantangan autentikasi definisi Anda dengan sesi awal yang berisi `challengeName: SRP_A challengeResult: true`
3. Setelah menerima masukan tersebut, fungsi Lambda Anda merespons dengan `challengeName: PASSWORD_VERIFIER, issueTokens: false, failAuthentication: false`.
4. Jika verifikasi kata sandi berhasil, Amazon Cognito memanggil fungsi Lambda Anda lagi dengan sesi baru yang berisi `challengeName: PASSWORD_VERIFIER challengeResult: true`
5. Untuk memulai tantangan kustom Anda, fungsi Lambda Anda merespons `challengeName: CUSTOM_CHALLENGE` dengan `issueTokens: false`, dan `failAuthentication: false` Jika Anda tidak ingin memulai alur autentikasi kustom dengan verifikasi kata sandi, Anda dapat memulai masuk dengan peta `AuthParameters` termasuk `CHALLENGE_NAME: CUSTOM_CHALLENGE`.
6. Putaran tantangan berulang sampai semua tantangan terjawab.

## Topik

- [Tentukan parameter pemicu Lambda tantangan Auth](#)
- [Tentukan contoh tantangan Auth](#)

## Tentukan parameter pemicu Lambda tantangan Auth

Permintaan yang diteruskan Amazon Cognito ke fungsi Lambda ini adalah kombinasi dari parameter di bawah ini dan parameter [umum yang ditambahkan Amazon](#) Cognito ke semua permintaan.

## JSON

```
{
  "request": {
    "userAttributes": {
      "string": "string",
      . . .
    },
    "session": [
      ChallengeResult,
      . . .
    ]
  }
}
```

```
    ],
    "clientMetadata": {
      "string": "string",
      . . .
    },
    "userNotFound": boolean
  },
  "response": {
    "challengeName": "string",
    "issueTokens": boolean,
    "failAuthentication": boolean
  }
}
```

## Tentukan parameter permintaan tantangan Auth

Saat Amazon Cognito memanggil fungsi Lambda Anda, Amazon Cognito menyediakan parameter berikut:

### userAttributes

Satu atau lebih pasangan nama-nilai yang mewakili atribut pengguna.

### userNotFound

Boolean yang diisi Amazon Cognito `PreventUserExistenceErrors` saat disetel `ENABLED` untuk klien kumpulan pengguna Anda. Nilai `true` berarti bahwa id pengguna (nama pengguna, alamat email, dan detail lainnya) tidak cocok dengan pengguna yang ada. Bila `PreventUserExistenceErrors` disetel ke `ENABLED`, layanan tidak menginformasikan aplikasi pengguna yang tidak ada. Kami menyarankan agar fungsi Lambda Anda mempertahankan pengalaman pengguna yang sama dan memperhitungkan latensi. Dengan cara ini, pemanggil tidak dapat mendeteksi perilaku yang berbeda ketika pengguna ada atau tidak ada.

### sesi


Array `ChallengeResult` elemen. Masing-masing berisi elemen-elemen berikut:

#### challengeName

Salah satu jenis tantangan

berikut: `CUSTOM_CHALLENGE`, `SRP_A`, `PASSWORD_VERIFIER`, `SMS_MFA`,  
`DEVICE_SRP_AUTH`, `DEVICE_PASSWORD_VERIFIER`, atau `ADMIN_NO_SRP_AUTH`.

Saat fungsi `define auth challenge` Anda mengeluarkan `PASSWORD_VERIFIER` tantangan bagi pengguna yang telah menyiapkan otentikasi multifaktor, Amazon Cognito menindaklanjutinya dengan sebuah tantangan. `SMS_MFA` Dalam fungsi Anda, sertakan penanganan untuk acara masukan dari `SMS_MFA` tantangan. Anda tidak perlu memanggil `SMS_MFA` tantangan dari fungsi `define auth challenge` Anda.

 **Important**

Saat fungsi Anda menentukan apakah pengguna telah berhasil diautentikasi dan Anda harus mengeluarkannya token, selalu periksa `challengeName` fungsi `define auth challenge` Anda dan verifikasi apakah itu cocok dengan nilai yang diharapkan.

### `challengeResult`

Atur ke `true` jika pengguna berhasil menyelesaikan tantangan, atau `false` untuk sebaliknya.

### `challengeMetadata`

Nama Anda untuk tantangan kustom. Digunakan hanya jika `challengeName` adalah `CUSTOM_CHALLENGE`.

### `clientMetadata`

Satu atau lebih pasangan nilai-kunci yang dapat Anda berikan sebagai masukan kustom ke fungsi Lambda yang Anda tentukan untuk pemacu menentukan tantangan autentikasi. Untuk meneruskan data ini ke fungsi Lambda Anda, Anda dapat menggunakan `ClientMetadata` parameter dalam operasi [AdminRespondToAuthChallenge](#) dan [RespondToAuthChallengeAPI](#). Permintaan yang memanggil fungsi `define auth challenge` tidak menyertakan data yang diteruskan dalam `ClientMetadata` parameter [AdminInitiateAuth](#) dan operasi [InitiateAuthAPI](#).

### Tentukan parameter respons tantangan Auth

Dalam respons, Anda dapat mengembalikan tahap berikutnya dari proses autentikasi.

### `challengeName`

String yang berisi nama tantangan berikutnya. Jika Anda ingin menyajikan tantangan baru bagi pengguna Anda, tentukan nama tantangan di sini.

## issueTokens

Jika Anda menentukan bahwa pengguna telah menyelesaikan tantangan otentikasi dengan cukup, atur `true`. Jika pengguna belum cukup memenuhi tantangan, atur `false`.

## failAuthentication

Jika Anda ingin mengakhiri proses otentikasi saat ini, atur `true`. Untuk melanjutkan proses otentikasi saat ini, atur `false`.

## Tentukan contoh tantangan Auth

Contoh ini mendefinisikan serangkaian tantangan untuk otentikasi dan mengeluarkan token hanya jika pengguna telah menyelesaikan semua tantangan dengan sukses.

## Node.js

```
const handler = async (event) => {
  if (
    event.request.session.length == 1 &&
    event.request.session[0].challengeName == "SRP_A"
  ) {
    event.response.issueTokens = false;
    event.response.failAuthentication = false;
    event.response.challengeName = "PASSWORD_VERIFIER";
  } else if (
    event.request.session.length == 2 &&
    event.request.session[1].challengeName == "PASSWORD_VERIFIER" &&
    event.request.session[1].challengeResult == true
  ) {
    event.response.issueTokens = false;
    event.response.failAuthentication = false;
    event.response.challengeName = "CUSTOM_CHALLENGE";
  } else if (
    event.request.session.length == 3 &&
    event.request.session[2].challengeName == "CUSTOM_CHALLENGE" &&
    event.request.session[2].challengeResult == true
  ) {
    event.response.issueTokens = false;
    event.response.failAuthentication = false;
    event.response.challengeName = "CUSTOM_CHALLENGE";
  } else if (
    event.request.session.length == 4 &&
```



```

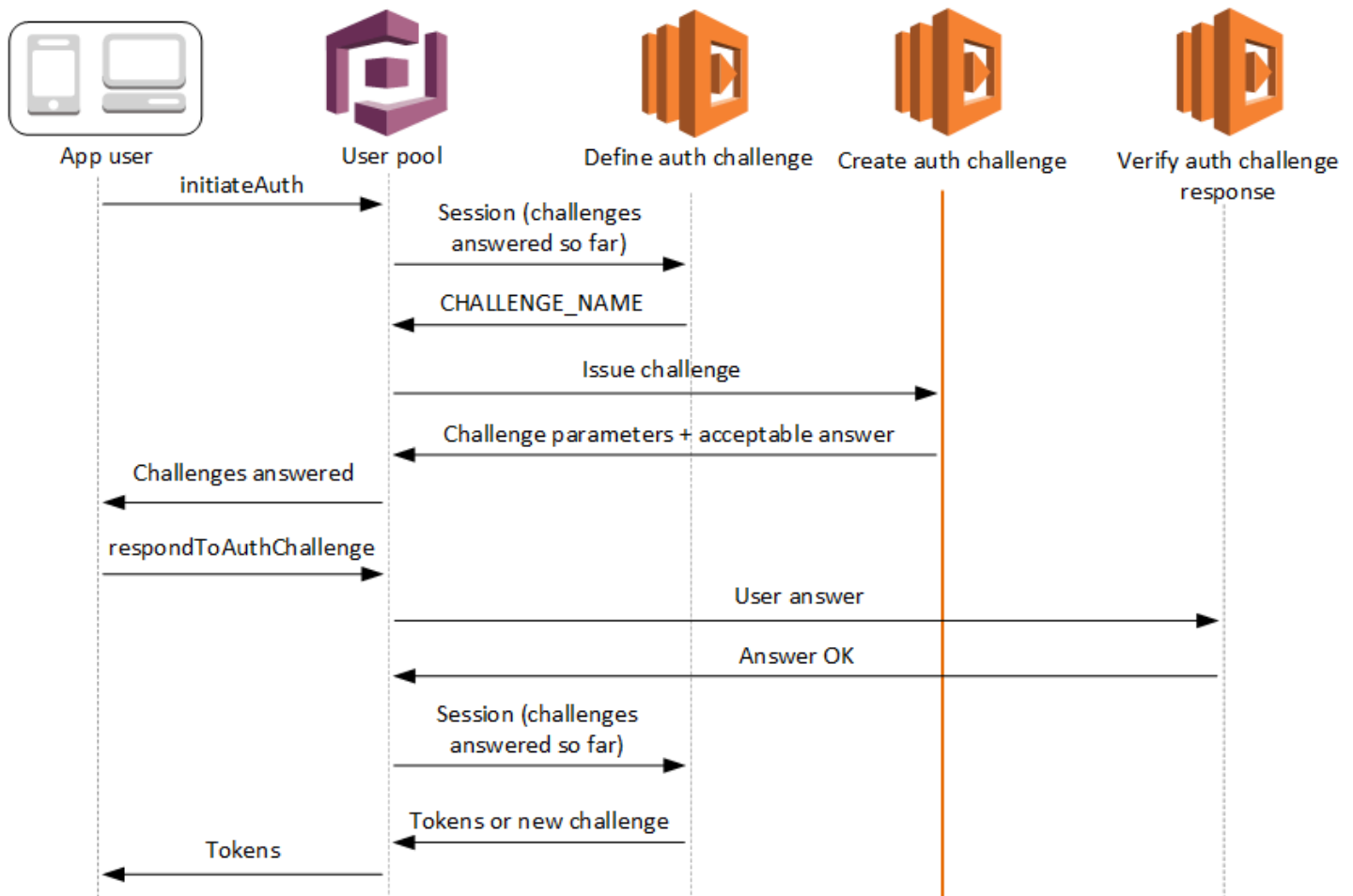
event.request.session[3].challengeName == "CUSTOM_CHALLENGE" &&
event.request.session[3].challengeResult == true
) {
  event.response.issueTokens = true;
  event.response.failAuthentication = false;
} else {
  event.response.issueTokens = false;
  event.response.failAuthentication = true;
}

return event;
};

export { handler }

```

## Membuat pemacu Lambda tantangan autentikasi



## Membuat tantangan autentikasi

Amazon Cognito memanggil pemacu ini setelah Menentukan Tantangan Autentikasi jika tantangan kustom telah ditetapkan sebagai bagian dari pemacu Menentukan Tantangan Autentikasi. Ini menciptakan [alur autentikasi kustom](#).

Pemacu Lambda ini dipanggil untuk membuat tantangan untuk disajikan kepada pengguna. Permintaan untuk pemacu Lambda ini mencakup `challengeName` dan `session`. `challengeName` adalah string dan merupakan nama dari tantangan berikutnya untuk pengguna. Nilai dari atribut ini diatur dalam pemacu Lambda Menentukan Tantangan Autentikasi.

Putaran tantangan akan berulang sampai semua tantangan terjawab.

### Topik

- [Membuat Parameter pemacu Lambda tantangan autentikasi](#)
- [Membuat Contoh Tantangan Autkasi](#)

## Membuat Parameter pemacu Lambda tantangan autentikasi

Permintaan yang diteruskan Amazon Cognito ke fungsi Lambda ini adalah kombinasi dari parameter di bawah ini dan [parameter umum](#) yang ditambahkan Amazon Cognito ke semua permintaan.

### JSON

```
{
  "request": {
    "userAttributes": {
      "string": "string",
      . . .
    },
    "challengeName": "string",
    "session": [
      ChallengeResult,
      . . .
    ],
    "clientMetadata": {
      "string": "string",
      . . .
    },
    "userNotFound": boolean
  }
```

```

    },
    "response": {
      "publicChallengeParameters": {
        "string": "string",
        . . .
      },
      "privateChallengeParameters": {
        "string": "string",
        . . .
      },
      "challengeMetadata": "string"
    }
  }
}

```

Buat parameter permintaan tantangan Auth

`userAttributes`

Satu atau lebih pasangan nilai-nama yang mewakili atribut pengguna.

`userNotFound`

Boolean ini diisi ketika `PreventUserExistenceErrors` diatur ke `ENABLED` untuk klien Kolam Pengguna Anda.

`challengeName`

Nama tantangan baru.

`sesi`

Elemen sesi adalah array elemen `ChallengeResult`, yang masing-masing berisi elemen berikut:

`challengeName`

Tipe tantangan. Salah satu dari: `"CUSTOM_CHALLENGE"`, `"PASSWORD_VERIFIER"`, `"SMS_MFA"`, `"DEVICE_SRP_AUTH"`, `"DEVICE_PASSWORD_VERIFIER"`, atau `"ADMIN_NO_SRP_AUTH"`.

`challengeResult`

Atur ke `true` jika pengguna berhasil menyelesaikan tantangan, atau `false` untuk sebaliknya.

## challengeMetadata

Nama Anda untuk tantangan kustom. Digunakan hanya jika challengeName adalah "CUSTOM\_CHALLENGE".

## clientMetadata

Satu atau lebih pasangan nilai-kunci yang dapat Anda berikan sebagai masukan kustom ke fungsi Lambda yang Anda tentukan untuk membuat pemicu tantangan autentikasi. Anda dapat menggunakan ClientMetadata parameter dalam tindakan [AdminRespondToAuthChallenge](#) dan [RespondToAuthChallenge](#) API untuk meneruskan data ini ke fungsi Lambda Anda. Permintaan yang memanggil fungsi create auth challenge tidak menyertakan data yang dilewatkan dalam ClientMetadata parameter dalam [AdminInitiateAuth](#) dan operasi [InitiateAuth](#) API.

## Buat parameter respons tantangan Auth

### publicChallengeParameters

Satu atau lebih pasangan nilai-kunci untuk digunakan oleh aplikasi klien dalam tantangan yang akan disajikan kepada pengguna. Parameter ini harus berisi semua informasi yang diperlukan untuk menyajikan tantangan kepada pengguna secara akurat.

### privateChallengeParameters

Parameter ini hanya digunakan oleh pemicu Lambda Memverifikasi Respons Tantangan Autentikasi. Parameter ini harus berisi semua informasi yang diperlukan untuk memvalidasi respons pengguna terhadap tantangan. Dengan kata lain, parameter publicChallengeParameters berisi pertanyaan yang disajikan kepada pengguna dan privateChallengeParameters berisi jawaban yang valid untuk pertanyaan tersebut.

## challengeMetadata

Nama Anda untuk tantangan kustom, jika ini adalah tantangan kustom.

## Membuat Contoh Tantangan Autkasi

Sebuah CAPTCHA dibuat sebagai tantangan bagi pengguna. URL untuk gambar CAPTCHA ditambahkan ke parameter tantangan publik sebagai "captchaUrl", dan jawaban yang diharapkan ditambahkan ke parameter tantangan privat.

## Node.js

```
const handler = async (event) => {
  if (event.request.challengeName !== "CUSTOM_CHALLENGE") {
    return event;
  }

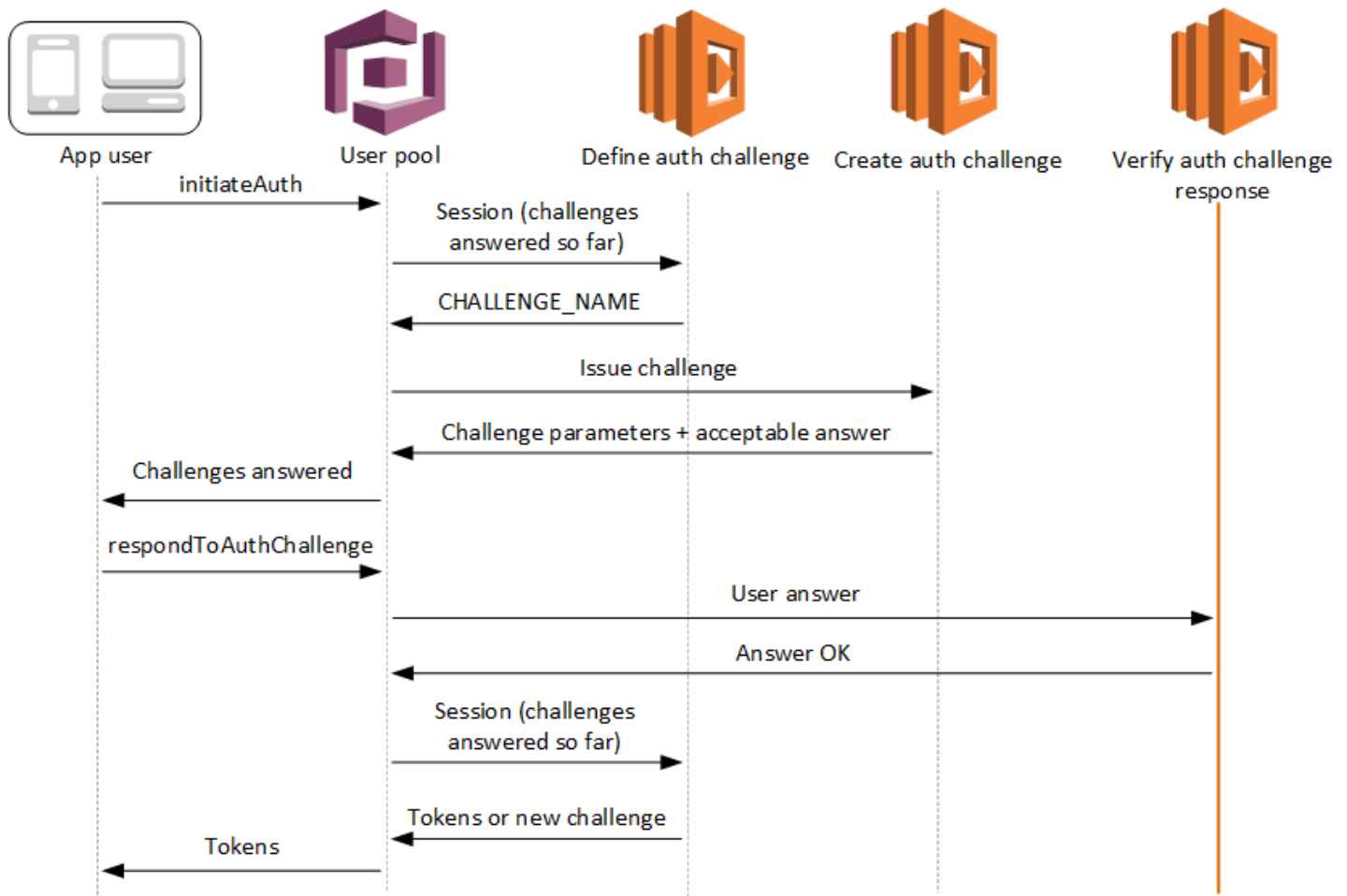
  if (event.request.session.length === 2) {
    event.response.publicChallengeParameters = {};
    event.response.privateChallengeParameters = {};
    event.response.publicChallengeParameters.captchaUrl = "url/123.jpg";
    event.response.privateChallengeParameters.answer = "5";
  }

  if (event.request.session.length === 3) {
    event.response.publicChallengeParameters = {};
    event.response.privateChallengeParameters = {};
    event.response.publicChallengeParameters.securityQuestion =
      "Who is your favorite team mascot?";
    event.response.privateChallengeParameters.answer = "Peccy";
  }

  return event;
};

export { handler }
```

## Memverifikasi pemacu Lambda respon tantangan autentikasi



## Memverifikasi respons tantangan autentikasi

Amazon Cognito memanggil pemacu ini apakah respons dari pengguna untuk Tantangan Autikasi kustom valid atau tidak. Ini adalah bagian dari kolam pengguna [alur autentikasi kustom](#).

Permintaan untuk pemacu ini berisi parameter `privateChallengeParameters` dan `challengeAnswer`. Pemacu Lambda Membuat Tantangan Autikasi, `privateChallengeParameters` dan berisi respons yang diharapkan dari pengguna. Parameter `challengeAnswer` berisi respons pengguna untuk tantangan.

Tanggapan berisi `answerCorrect` atribut. Jika pengguna berhasil menyelesaikan tantangan, Amazon Cognito menetapkan nilai atribut ke `true`. Jika pengguna tidak berhasil menyelesaikan tantangan, Amazon Cognito menetapkan nilainya `false`.

Putaran tantangan terjawab semua tantangan.

## Topik

- [Memverifikasi parameter Lambda tantangan Auttikasi](#)
- [Verifikasi contoh respons tantangan Auth](#)

### Memverifikasi parameter Lambda tantangan Auttikasi

Permintaan yang diteruskan Amazon Cognito ke fungsi Lambda ini adalah kombinasi dari parameter di bawah ini dan [parameter umum](#) yang ditambahkan Amazon Cognito ke semua permintaan.

### JSON

```
{
  "request": {
    "userAttributes": {
      "string": "string",
      . . .
    },
    "privateChallengeParameters": {
      "string": "string",
      . . .
    },
    "challengeAnswer": "string",
    "clientMetadata": {
      "string": "string",
      . . .
    },
    "userNotFound": boolean
  },
  "response": {
    "answerCorrect": boolean
  }
}
```

### Verifikasi parameter permintaan tantangan Auth

#### userAttributes

Parameter ini berisi satu atau lebih pasangan nilai-nama yang mewakili atribut pengguna.

## userNotFound

Ketika Amazon Cognito menetapkan `PreventUserExistenceErrors` `ENABLED` untuk klien kumpulan pengguna Anda, Amazon Cognito mengisi Boolean ini.

## privateChallengeParameters

Parameter ini berasal dari pemicu `Create Auth Challenge`. Untuk menentukan apakah pengguna melewati tantangan, Amazon Cognito membandingkan parameter dengan `challengeAnswer` pengguna.

Parameter ini berisi semua informasi yang diperlukan untuk memvalidasi respons dari tantangan. Informasi itu mencakup pertanyaan yang disajikan Amazon Cognito kepada pengguna (`publicChallengeParameters`), dan jawaban yang valid untuk pertanyaan (`privateChallengeParameters`). Hanya pemicu `Lambda Respons Tantangan Autentikasi`.

## challengeAnswer

Nilai parameter ini adalah jawaban dari respons dari tantangan tersebut.

## clientMetadata

Parameter ini berisi satu atau lebih pasangan nilai-kunci yang dapat Anda berikan sebagai masukan kustom ke fungsi `Lambda` untuk pemicu memverifikasi tantangan autentikasi. Untuk meneruskan data ini ke fungsi `Lambda` Anda, gunakan `ClientMetadata` parameter dalam operasi [AdminRespondToAuthChallenge](#) dan [RespondToAuthChallenge](#) API. Amazon Cognito tidak menyertakan data dari `ClientMetadata` parameter dalam [AdminInitiateAuth](#) dan operasi [InitiateAuth](#) API dalam permintaan yang diteruskan ke fungsi tantangan autentikasi verifikasi.

## Verifikasi parameter respons tantangan Auth

### answerCorrect

Jika pengguna berhasil menyelesaikan tantangan, Amazon Cognito menetapkan parameter ini. `true` Jika pengguna tidak berhasil menyelesaikan tantangan, Amazon Cognito menetapkan parameter ke `false`.

## Verifikasi contoh respons tantangan Auth

Dalam contoh ini, fungsi `Lambda` memeriksa apakah respons pengguna terhadap tantangan cocok dengan respons yang diharapkan. Jika respons dari pengguna sesuai dengan respons yang diharapkan, Amazon Cognito akan `answerCorrect`. `true`



## Node.js

```
const handler = async (event) => {
  if (
    event.request.privateChallengeParameters.answer ==
    event.request.challengeAnswer
  ) {
    event.response.answerCorrect = true;
  } else {
    event.response.answerCorrect = false;
  }

  return event;
};

export { handler };
```

## Pemicu Lambda generasi pra token

Karena Amazon Cognito memanggil pemicu ini sebelum pembuatan token, Anda dapat menyesuaikan klaim dalam token kumpulan pengguna. Dengan fitur Dasar dari versi satu atau peristiwa pemicu pembuatan token V1\_0 pra, Anda dapat menyesuaikan token identitas (ID). Di kumpulan pengguna dengan [fitur keamanan lanjutan](#) yang aktif, Anda dapat menghasilkan versi 2 atau V2\_0 memicu peristiwa dengan kustomisasi token akses.

Amazon Cognito mengirimkan V1\_0 acara sebagai permintaan ke fungsi Anda dengan data yang akan ditulis ke token ID. V2\_0 Peristiwa adalah permintaan tunggal dengan data yang akan ditulis Amazon Cognito ke identitas dan token akses. Untuk menyesuaikan kedua token, Anda harus memperbarui fungsi Anda untuk menggunakan versi pemicu terbaru, dan mengirim data untuk kedua token dalam respons yang sama.

Pemicu Lambda ini dapat menambah, menghapus, dan memodifikasi beberapa klaim dalam identitas dan token akses sebelum Amazon Cognito menerbitkannya ke aplikasi Anda. Untuk menggunakan fitur ini, kaitkan fungsi Lambda dari konsol kumpulan pengguna Amazon Cognito atau perbarui LambdaConfig kumpulan pengguna Anda melalui AWS Command Line Interface (.AWS CLI

## Versi acara

Kumpulan pengguna Anda dapat mengirimkan versi berbeda dari peristiwa pemicu pembuatan token pra ke fungsi Lambda Anda. V1\_0Pemicu memberikan parameter untuk modifikasi token ID. V2\_0Pemicu memberikan parameter untuk yang berikut ini.

1. Fungsi V1\_0 pemicu.
2. Kemampuan untuk menyesuaikan token akses.
3. Kemampuan untuk meneruskan tipe data kompleks ke ID dan mengakses nilai klaim token:
  - String
  - Number
  - Boolean
  - Array string, angka, boolean, atau kombinasi dari semua ini
  - JSON

### Note

Dalam token ID, Anda dapat mengisi objek kompleks dengan nilai klaim ke `phone_number_verified`, `email_verifiedupdated_at`, dan `address`.

Kumpulan pengguna mengirimkan V1\_0 acara secara default. Untuk mengonfigurasi kumpulan pengguna Anda untuk mengirim V2\_0 peristiwa, pilih versi peristiwa Pemicu dari fitur Dasar +penyesuaian token akses saat Anda mengonfigurasi pemicu di konsol Amazon Cognito. Anda juga dapat mengatur nilai `LambdaVersion` dalam `LambdaConfig` parameter dalam permintaan `UpdateUserPool` atau `CreateUserPool` API. Biaya tambahan berlaku untuk mengakses kustomisasi token dengan V2\_0 acara. Untuk informasi selengkapnya, lihat [Harga Amazon Cognito](#).

## Klaim dan cakupan yang dikecualikan

Amazon Cognito membatasi klaim dan cakupan yang dapat Anda tambahkan, modifikasi, atau tekan dalam token akses dan identitas. Jika fungsi Lambda Anda mencoba menetapkan nilai untuk salah satu klaim ini, Amazon Cognito mengeluarkan token dengan nilai klaim asli, jika ada dalam permintaan.

## Klaim bersama

- `acr`
- `amr`
- `at_hash`
- `auth_time`
- `azp`
- `exp`
- `iat`
- `iss`
- `jti`
- `nbf`
- `nonce`
- `origin_jti`
- `sub`
- `token_use`

## Klaim token ID

- `identities`
- `aud`
- `cognito:username`

## Klaim token akses

- `username`
- `client_id`
- `scope`

### Note

Anda dapat mengubah cakupan dalam token akses dengan `scopesToAdd` dan nilai `scopesToSuppress` respons, tetapi Anda tidak dapat memodifikasi `scope` klaim secara

langsung. Anda tidak dapat menambahkan cakupan yang dimulai `aws.cognito.signin.user.admin` cadangan kumpulan pengguna.

- `device_key`
- `event_id`
- `version`

Anda tidak dapat menambahkan atau mengganti klaim dengan awalan berikut tetapi Anda dapat menekannya, atau mencegahnya muncul di token.

- `dev:`
- `cognito:`

Anda dapat menambahkan aud klaim ke token akses, tetapi nilainya harus sesuai dengan ID klien aplikasi sesi saat ini. Anda dapat memperoleh ID klien dalam acara permintaan dari `event.callerContext.clientId`.

## Menyesuaikan token identitas

Dengan pemicu Lambda pembuatan token pra, Anda dapat menyesuaikan konten token identitas (ID) dari kumpulan pengguna Anda. Token ID menyediakan atribut pengguna dari sumber identitas terpercaya untuk login ke web atau aplikasi seluler. Untuk informasi selengkapnya tentang token ID, lihat [Menggunakan token ID](#).

Penggunaan pemicu Lambda generasi pra token dengan token ID meliputi yang berikut ini.

- Buat perubahan saat runtime ke peran IAM yang diminta pengguna Anda dari kumpulan identitas.
- Tambahkan atribut pengguna dari sumber eksternal.
- Tambahkan atau ganti nilai atribut pengguna yang ada.
- Menekan pengungkapan atribut pengguna yang, karena cakupan resmi pengguna Anda dan akses baca ke atribut yang Anda berikan kepada klien aplikasi, akan diteruskan ke aplikasi Anda.

## Menyesuaikan token akses

Dengan pemicu Lambda generasi pra token, Anda dapat menyesuaikan konten token akses dari kumpulan pengguna Anda. Token akses memberi wewenang kepada pengguna untuk mengambil

informasi dari sumber daya yang dilindungi akses seperti operasi API resmi token Amazon Cognito dan API pihak ketiga. Meskipun Anda dapat menghasilkan token akses untuk otorisasi machine-to-machine (M2M) dengan Amazon Cognito dengan hibah kredensial klien, permintaan M2M tidak menjalankan fungsi pemicu pembuatan token pra dan tidak dapat mengeluarkan token akses yang disesuaikan. Untuk informasi selengkapnya tentang token akses, lihat [Menggunakan token akses](#).

Penggunaan pemicu Lambda generasi pra token dengan token akses meliputi yang berikut ini.

- Tambahkan atau tekan cakupan OAuth 2.0 dalam klaim. scope Misalnya, Anda dapat menambahkan cakupan ke token akses yang dihasilkan dari autentikasi API kumpulan pengguna Amazon Cognito, yang hanya menetapkan cakupan. `aws.cognito.signin.user.admin`
- Ubah keanggotaan pengguna dalam grup kumpulan pengguna.
- Tambahkan klaim yang belum ada di token akses Amazon Cognito.
- Menekan pengungkapan klaim yang seharusnya diteruskan ke aplikasi Anda.

Untuk mendukung penyesuaian akses di kumpulan pengguna Anda, Anda harus mengonfigurasi kumpulan pengguna untuk menghasilkan versi terbaru dari permintaan pemicu. Perbarui kumpulan pengguna Anda seperti yang ditunjukkan dalam prosedur berikut.

## AWS Management Console

Untuk mendukung kustomisasi token akses dalam pemicu Lambda generasi pra token

1. Buka [konsol Amazon Cognito](#), lalu pilih Kumpulan Pengguna.
2. Pilih kolam pengguna yang ada dari daftar, atau [buat kolam pengguna](#).
3. Jika Anda belum melakukannya, aktifkan [fitur keamanan lanjutan](#) dari tab Integrasi aplikasi.
4. Pilih tab Properti kumpulan pengguna dan temukan pemicu Lambda.
5. Tambahkan atau edit pemicu pembuatan Pra token.
6. Pilih fungsi Lambda di bawah fungsi Tetapkan Lambda.
7. Pilih versi acara Trigger dari fitur Basic + kustomisasi token akses. Pengaturan ini memperbarui parameter permintaan yang dikirimkan Amazon Cognito ke fungsi Anda untuk menyertakan bidang untuk kustomisasi token akses.

## User pools API

Untuk mendukung kustomisasi token akses dalam pemicu Lambda generasi pra token

Buat permintaan [CreateUserPool](#) atau [UpdateUserPool](#) API. Anda harus menentukan nilai untuk semua parameter yang tidak ingin disetel ke nilai default. Untuk informasi selengkapnya, lihat [Memperbarui konfigurasi kumpulan pengguna](#).

Sertakan konten berikut dalam `LambdaVersion` parameter permintaan Anda.

`LambdaVersion` Nilai `V2_0` menyebabkan kumpulan pengguna Anda menambahkan parameter untuk kustomisasi token akses. Untuk memanggil versi fungsi tertentu, gunakan ARN fungsi Lambda dengan versi fungsi sebagai nilai. `LambdaArn`

```
"PreTokenGenerationConfig": {
  "LambdaArn": "arn:aws:lambda:us-west-2:123456789012:function:MyFunction",
  "LambdaVersion": "V2_0"
},
```

## Topik

- [Sumber pemicu Lambda generasi pra token](#)
- [Parameter pemicu Lambda generasi pra token](#)
- [Contoh peristiwa pemicu pra token versi dua: Menambahkan dan menekan klaim, cakupan, dan grup](#)
- [Contoh acara pembuatan pra token versi dua: Tambahkan klaim dengan objek kompleks](#)
- [Versi acara pembuatan token pra satu contoh: Tambahkan klaim baru dan tekan klaim yang ada](#)
- [Versi acara pembuatan pra token satu contoh: Ubah keanggotaan grup pengguna](#)

## Sumber pemicu Lambda generasi pra token

Nilai triggerSource	Peristiwa
TokenGeneration_HostedAuth	Dipanggil selama autentikasi dari halaman masuk UI yang dihosting Amazon Cognito.
TokenGeneration_Authentication	Dipanggil setelah alur autentikasi pengguna selesai.
TokenGeneration_NewPassword Challenge	Dipanggil setelah pengguna dibuat oleh admin. Alur ini dipanggil ketika pengguna harus mengubah kata sandi sementara.

Nilai triggerSource	Peristiwa
TokenGeneration_AuthenticateDevice	Dipanggil pada akhir autentikasi perangkat pengguna.
TokenGeneration_RefreshTokens	Dipanggil saat pengguna mencoba menyegarkan identitas dan mengakses token.

## Parameter pemicu Lambda generasi pra token

Permintaan yang diteruskan Amazon Cognito ke fungsi Lambda ini adalah kombinasi dari parameter di bawah ini dan parameter [umum yang ditambahkan Amazon Cognito](#) ke semua permintaan. Saat Anda menambahkan pemicu Lambda generasi pra token ke kumpulan pengguna Anda, Anda dapat memilih versi pemicu. Versi ini menentukan apakah Amazon Cognito meneruskan permintaan ke fungsi Lambda Anda dengan parameter tambahan untuk kustomisasi token akses.

### Version 1

Token versi 1 dapat mengatur keanggotaan grup, peran IAM, dan klaim baru dalam token ID.

```
{
  "request": {
    "userAttributes": {"string": "string"},
    "groupConfiguration": {
      "groupsToOverride": [
        "string",
        "string"
      ],
      "iamRolesToOverride": [
        "string",
        "string"
      ],
      "preferredRole": "string"
    },
    "clientMetadata": {"string": "string"}
  },
  "response": {
    "claimsOverrideDetails": {
      "claimsToAddOrOverride": {"string": "string"},
      "claimsToSuppress": [
        "string",
```

```

        "string"
    ],
    "groupOverrideDetails": {
        "groupsToOverride": [
            "string",
            "string"
        ],
        "iamRolesToOverride": [
            "string",
            "string"
        ],
        "preferredRole": "string"
    }
}
}
}
}

```

## Version 2

Peristiwa permintaan versi 2 menambahkan bidang yang menyesuaikan token akses. Hal ini juga menambahkan dukungan untuk tipe `claimsToOverride` data yang kompleks dalam objek respon. Fungsi Lambda Anda dapat mengembalikan jenis data berikut dalam nilai `claimsToOverride`

- String
- Number
- Boolean
- Array string, angka, boolean, atau kombinasi dari semua ini
- JSON

```

{
  "request": {
    "userAttributes": {
      "string": "string"
    },
    "scopes": ["string", "string"],
    "groupConfiguration": {
      "groupsToOverride": ["string", "string"],
      "iamRolesToOverride": ["string", "string"],
      "preferredRole": "string"
    }
  }
}

```



```

    },
    "clientMetadata": {
      "string": "string"
    }
  },
  "response": {
    "claimsAndScopeOverrideDetails": {
      "idTokenGeneration": {
        "claimsToAddOrOverride": {
          "string": [accepted datatype]
        },
        "claimsToSuppress": ["string", "string"]
      },
      "accessTokenGeneration": {
        "claimsToAddOrOverride": {
          "string": [accepted datatype]
        },
        "claimsToSuppress": ["string", "string"],
        "scopesToAdd": ["string", "string"],
        "scopesToSuppress": ["string", "string"]
      },
      "groupOverrideDetails": {
        "groupsToOverride": ["string", "string"],
        "iamRolesToOverride": ["string", "string"],
        "preferredRole": "string"
      }
    }
  }
}

```

## Parameter permintaan pembuatan pra token

Nama	Penjelasan	Versi acara pemicu minimum
userAttributes	Atribut profil pengguna Anda di kumpulan pengguna Anda.	1
groupConfiguration	Objek masukan yang berisi konfigurasi grup saat ini. Objek termasuk <code>groupsToOverride</code> , <code>iamRolesToOverride</code> , dan <code>preferredRole</code> .	1

Nama	Penjelasan	Versi acara pemicu minimum
groupsToOverride	<a href="#">Kelompok kumpulan pengguna</a> yang menjadi anggota pengguna Anda.	1
iamRolesToAssign	Anda dapat mengaitkan grup kumpulan pengguna dengan peran AWS Identity and Access Management (IAM). Elemen ini adalah daftar semua peran IAM dari grup tempat pengguna Anda menjadi anggota.	1
preferredRole	Anda dapat menetapkan <a href="#">prioritas untuk grup</a> kumpulan pengguna. Elemen ini berisi nama peran IAM dari grup dengan presendensi tertinggi dalam elemen. groupsToOverride	1
clientMetadata	<p>Satu atau beberapa pasangan nilai kunci yang dapat Anda tentukan dan berikan sebagai input khusus ke fungsi Lambda untuk pemicu pembuatan token pra.</p> <p>Untuk meneruskan data ini ke fungsi Lambda Anda, gunakan ClientMetadata parameter dalam operasi <a href="#">AdminRespondToAuthChallenge</a> dan <a href="#">RespondToAuthChallenge</a> API. Amazon Cognito tidak menyertakan data dari ClientMetadata parameter dalam <a href="#">AdminInitiateAuth</a> dan operasi <a href="#">InitiateAuth</a> API dalam permintaan yang diteruskan ke fungsi pembuatan token pra.</p>	1
ruang lingkup	Cakupan OAuth 2.0 pengguna Anda. Cakupan yang ada dalam token akses adalah standar kumpulan pengguna dan cakupan kustom yang diminta pengguna, dan bahwa Anda mengizinkan klien aplikasi Anda untuk menerbitkan.	2

## Parameter respons pembuatan pra token

Nama	Penjelasan	Versi acara pemicu minimum
<code>claimsOverrideDetails</code>	Sebuah wadah untuk semua elemen dalam peristiwa <code>V1_0</code> pemicu.	1
<code>claimsAndScopeOverrideDetails</code>	Sebuah wadah untuk semua elemen dalam peristiwa <code>V2_0</code> pemicu.	2
<code>idTokenGeneration</code>	Klaim yang ingin Anda ganti, tambahkan, atau tekan di token ID pengguna Anda. Nilai kustomisasi token induk ke ID ini hanya muncul di acara versi 2, tetapi elemen anak muncul di acara versi 1.	2
<code>accessTokenGeneration</code>	Klaim dan cakupan yang ingin Anda timpa, tambahkan, atau tekan di token akses pengguna Anda. Induk ini untuk mengakses nilai kustomisasi token hanya muncul di acara versi 2.	2
<code>claimsToAddOrOverride</code>	<p>Peta dari satu atau lebih klaim dan nilainya yang ingin Anda tambahkan atau modifikasi. Untuk klaim terkait grup, gunakan <code>groupOverrideDetails</code> sebagai gantinya.</p> <p>Dalam acara versi 2, elemen ini muncul di bawah keduanya <code>accessTokenGeneration</code> dan <code>idTokenGeneration</code> .</p>	1 <sup>*</sup>
<code>claimsToSuppress</code>	<p>Daftar klaim yang ingin Anda tekan oleh Amazon Cognito. Jika fungsi Anda menekan dan mengganti nilai klaim, Amazon Cognito akan menekan klaim tersebut.</p> <p>Dalam acara versi 2, elemen ini muncul di bawah keduanya <code>accessTokenGeneration</code> dan <code>idTokenGeneration</code> .</p>	1

Nama	Penjelasan	Versi acara pemicu minimum
groupOverrideDetails	<p>Objek output yang berisi konfigurasi grup saat ini. Objek termasuk <code>groupsToOverride</code>, <code>iamRolesToOverride</code>, dan <code>preferredRole</code>.</p> <p>Fungsi Anda menggantikan <code>groupOverrideDetails</code> objek dengan objek yang Anda berikan. Jika Anda memberikan objek kosong atau null dalam respons, Amazon Cognito akan menekan grup. Untuk menjaga konfigurasi grup yang ada tetap sama, salin nilai <code>groupConfiguration</code> objek permintaan ke <code>groupOverrideDetails</code> objek dalam respons. Kemudian berikan kembali ke layanan.</p> <p>ID Amazon Cognito dan token akses keduanya berisi klaim <code>cognito:groups</code>. <code>groupOverrideDetails</code> Objek Anda menggantikan <code>cognito:groups</code> klaim dalam token akses dan token ID.</p>	1
scopesToAdd	Daftar cakupan OAuth 2.0 yang ingin Anda tambahkan ke scope klaim di token akses pengguna Anda. Anda tidak dapat menambahkan nilai cakupan yang berisi satu atau beberapa karakter ruang kosong.	2
scopesToSuppress	Daftar cakupan OAuth 2.0 yang ingin Anda hapus dari scope klaim di token akses pengguna Anda.	2

\* Objek respons ke acara versi 1 dapat mengembalikan string. Objek respons terhadap peristiwa versi 2 dapat mengembalikan [objek yang kompleks](#).

Contoh peristiwa pemicu pra token versi dua: Menambahkan dan menekan klaim, cakupan, dan grup

Contoh ini membuat modifikasi berikut pada token pengguna.

1. Menetapkan mereka `family_name` sebagai Doe dalam token ID.

2. Mencegah email dan phone\_number klaim agar tidak muncul di token ID.
3. Menetapkan cognito:roles klaim token ID mereka ke "arn:aws:iam::123456789012:role\sns\_callerA", "arn:aws:iam::123456789012:role\sns\_callerC", "arn:aws:iam::123456789012:role\sns\_callerB".
4. Menetapkan cognito:preferred\_role klaim token ID mereka ke arn:aws:iam::123456789012:role/sns\_caller.
5. Menambahkan cakupan openid, email, dan solar-system-data/asteroids.add ke token akses.
6. Menekan ruang lingkup phone\_number dan aws.cognito.signin.user.admin dari token akses. Penghapusan phone\_number mencegah pengambilan nomor telepon pengguna dari userInfo. Penghapusan aws.cognito.signin.user.admin mencegah permintaan API oleh pengguna untuk membaca dan memodifikasi profil mereka sendiri dengan API kumpulan pengguna Amazon Cognito.

#### Note

Penghapusan phone\_number dari cakupan hanya mencegah pengambilan nomor telepon pengguna jika cakupan yang tersisa dalam token akses termasuk openid dan setidaknya satu ruang lingkup standar lagi. Untuk informasi selengkapnya, lihat [Tentang cakupan](#).

7. Menetapkan cognito:groups klaim ID dan token akses mereka ke "new-group-A", "new-group-B", "new-group-C".

## JavaScript

```
export const handler = function(event, context) {
  event.response = {
    "claimsAndScopeOverrideDetails": {
      "idTokenGeneration": {
        "claimsToAddOrOverride": {
          "family_name": "Doe"
        },
        "claimsToSuppress": [
          "email",
          "phone_number"
        ]
      },
      "accessTokenGeneration": {
```

```

    "scopesToAdd": [
      "openid",
      "email",
      "solar-system-data/asteroids.add"
    ],
    "scopesToSuppress": [
      "phone_number",
      "aws.cognito.signin.user.admin"
    ]
  },
  "groupOverrideDetails": {
    "groupsToOverride": [
      "new-group-A",
      "new-group-B",
      "new-group-C"
    ],
    "iamRolesToOverride": [
      "arn:aws:iam::123456789012:role/new_roleA",
      "arn:aws:iam::123456789012:role/new_roleB",
      "arn:aws:iam::123456789012:role/new_roleC"
    ],
    "preferredRole": "arn:aws:iam::123456789012:role/new_role",
  }
}
};
// Return to Amazon Cognito
context.done(null, event);
};

```

Amazon Cognito meneruskan informasi peristiwa ke fungsi Lambda Anda. Fungsi kemudian mengembalikan objek acara yang sama ke Amazon Cognito, dengan perubahan apa pun dalam respons. Di konsol Lambda, Anda dapat mengatur peristiwa pengujian dengan data yang relevan dengan pemicu Lambda Anda. Berikut ini adalah peristiwa pengujian untuk sampel kode ini:

## JSON

```

{
  "version": "2",
  "triggerSource": "TokenGeneration_Authentication",
  "region": "us-east-1",
  "userPoolId": "us-east-1_EXAMPLE",
  "userName": "JaneDoe",

```

```
"callerContext": {
  "awsSdkVersion": "aws-sdk-unknown-unknown",
  "clientId": "1example23456789"
},
"request": {
  "userAttributes": {
    "sub": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "cognito:user_status": "CONFIRMED",
    "email_verified": "true",
    "phone_number_verified": "true",
    "phone_number": "+12065551212",
    "family_name": "Zoe",
    "email": "Jane.Doe@example.com"
  },
  "groupConfiguration": {
    "groupsToOverride": ["group-1", "group-2", "group-3"],
    "iamRolesToOverride": ["arn:aws:iam::123456789012:role/sns_caller1",
"arn:aws:iam::123456789012:role/sns_caller2", "arn:aws:iam::123456789012:role/
sns_caller3"],
    "preferredRole": ["arn:aws:iam::123456789012:role/sns_caller"]
  },
  "scopes": [
    "aws.cognito.signin.user.admin", "openid", "email", "phone"
  ]
},
"response": {
  "claimsAndScopeOverrideDetails": []
}
}
```

## Contoh acara pembuatan pra token versi dua: Tambahkan klaim dengan objek kompleks

Contoh ini membuat modifikasi berikut pada token pengguna.

1. Menambahkan klaim nomor, string, boolean, dan tipe JSON ke token ID. Ini adalah satu-satunya perubahan yang disediakan oleh peristiwa pemicu versi dua untuk token ID.
2. Menambahkan klaim nomor, string, boolean, dan tipe JSON ke token akses.
3. Menambahkan tiga cakupan ke token akses.
4. Menekan email dan sub klaim dalam ID dan token akses.

## 5. Menekan `aws.cognito.signin.user.admin` ruang lingkup dalam token akses.

### JavaScript

```
export const handler = function(event, context) {

  var scopes = ["MyAPI.read", "MyAPI.write", "MyAPI.admin"]
  var claims = {}
  claims["aud"]= event.callerContext.clientId;
  claims["booleanTest"] = false;
  claims["longTest"] = 9223372036854775807;
  claims["exponentTest"] = 1.7976931348623157E308;
  claims["ArrayTest"] = ["test", 9223372036854775807, 1.7976931348623157E308,
true];
  claims["longStringTest"] = "\\{\\
  \\\"first_json_block\\\": \\{\\
    \\\"key_A\\\": \\\"value_A\\\",\\
    \\\"key_B\\\": \\\"value_B\\\"\\
  \\},\\
  \\\"second_json_block\\\": \\{\\
    \\\"key_C\\\": \\{\\
      \\\"subkey_D\\\": [\\
        \\\"value_D\\\",\\
        \\\"value_E\\\"\\
      ],\\
      \\\"subkey_F\\\": \\\"value_F\\\"\\
    \\},\\
    \\\"key_G\\\": \\\"value_G\\\"\\
  \\}\\
  \\}\";
  claims["jsonTest"] = {
    "first_json_block": {
      "key_A": "value_A",
      "key_B": "value_B"
    },
    "second_json_block": {
      "key_C": {
        "subkey_D": [
          "value_D",
          "value_E"
        ],
        "subkey_F": "value_F"
      }
    }
  },
```



```

    "key_G": "value_G"
  }
};
event.response = {
  "claimsAndScopeOverrideDetails": {
    "idTokenGeneration": {
      "claimsToAddOrOverride": claims,
      "claimsToSuppress": ["email","sub"]
    },
    "accessTokenGeneration": {
      "claimsToAddOrOverride": claims,
      "claimsToSuppress": ["email","sub"],
      "scopesToAdd": scopes,
      "scopesToSuppress": ["aws.cognito.signin.user.admin"]
    }
  }
};
console.info("EVENT response\n" + JSON.stringify(event, (_, v) => typeof v ===
'bigint' ? v.toString() : v, 2))
console.info("EVENT response size\n" + JSON.stringify(event, (_, v) => typeof v
=== 'bigint' ? v.toString() : v).length)
// Return to Amazon Cognito
context.done(null, event);
};

```

Amazon Cognito meneruskan informasi peristiwa ke fungsi Lambda Anda. Fungsi kemudian mengembalikan objek acara yang sama ke Amazon Cognito, dengan perubahan apa pun dalam respons. Di konsol Lambda, Anda dapat mengatur peristiwa pengujian dengan data yang relevan dengan pemicu Lambda Anda. Berikut ini adalah peristiwa pengujian untuk sampel kode ini:

## JSON

```

{
  "version": "2",
  "triggerSource": "TokenGeneration_HostedAuth",
  "region": "us-west-2",
  "userPoolId": "us-west-2_EXAMPLE",
  "userName": "JaneDoe",
  "callerContext": {
    "awsSdkVersion": "aws-sdk-unknown-unknown",
    "clientId": "1example23456789"
  }
},

```

```
"request": {
  "userAttributes": {
    "sub": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "cognito:user_status": "CONFIRMED"
    "email_verified": "true",
    "phone_number_verified": "true",
    "phone_number": "+12065551212",
    "email": "Jane.Doe@example.com"
  },
  "groupConfiguration": {
    "groupsToOverride": ["group-1", "group-2", "group-3"],
    "iamRolesToOverride": ["arn:aws:iam::123456789012:role/sns_caller1"],
    "preferredRole": ["arn:aws:iam::123456789012:role/sns_caller1"]
  },
  "scopes": [
    "aws.cognito.signin.user.admin",
    "phone",
    "openid",
    "profile",
    "email"
  ]
},
"response": {
  "claimsAndScopeOverrideDetails": []
}
}
```

Versi acara pembuatan token pra satu contoh: Tambahkan klaim baru dan tekan klaim yang ada

Contoh ini menggunakan peristiwa pemicu versi 1 dengan fungsi Lambda pembuatan token pra untuk menambahkan klaim baru dan menekan klaim yang ada.

Node.js

```
const handler = async (event) => {
  event.response = {
    claimsOverrideDetails: {
      claimsToAddOrOverride: {
        my_first_attribute: "first_value",
        my_second_attribute: "second_value",
      },

```

```
    claimsToSuppress: ["email"],
  },
};

return event;
};

export { handler };
```

Amazon Cognito meneruskan informasi peristiwa ke fungsi Lambda Anda. Fungsi kemudian mengembalikan objek acara yang sama ke Amazon Cognito, dengan perubahan apa pun dalam respons. Di konsol Lambda, Anda dapat mengatur peristiwa pengujian dengan data yang relevan dengan pemicu Lambda Anda. Berikut ini adalah peristiwa pengujian untuk contoh kode ini: Karena contoh kode tidak memproses parameter permintaan apa pun, Anda dapat menggunakan peristiwa pengujian dengan permintaan kosong. Untuk informasi selengkapnya tentang parameter permintaan umum, lihat [Acara pemicu Lambda kumpulan pengguna](#).

## JSON

```
{
  "request": {},
  "response": {}
}
```

## Versi acara pembuatan pra token satu contoh: Ubah keanggotaan grup pengguna

Contoh ini menggunakan peristiwa pemicu versi 1 dengan fungsi Lambda pembuatan token pra untuk memodifikasi keanggotaan grup pengguna.

## Node.js

```
const handler = async (event) => {
  event.response = {
    claimsOverrideDetails: {
      groupOverrideDetails: {
        groupsToOverride: ["group-A", "group-B", "group-C"],
        iamRolesToOverride: [
          "arn:aws:iam::XXXXXXXXXXXX:role/sns_callerA",
          "arn:aws:iam::XXXXXXXXXXXX:role/sns_callerB",
          "arn:aws:iam::XXXXXXXXXXXX:role/sns_callerC",
        ],
      },
    },
  };
};
```

```
    ],
    preferredRole: "arn:aws:iam::XXXXXXXXXXXX:role/sns_caller",
  },
},
};

return event;
};

export { handler };
```

Amazon Cognito meneruskan informasi peristiwa ke fungsi Lambda Anda. Fungsi kemudian mengembalikan objek acara yang sama ke Amazon Cognito, dengan perubahan apa pun dalam respons. Di konsol Lambda, Anda dapat mengatur peristiwa pengujian dengan data yang relevan dengan pemicu Lambda Anda. Berikut ini adalah peristiwa pengujian untuk sampel kode ini:

## JSON

```
{
  "request": {},
  "response": {}
}
```

## Memigrasi pengguna pemicu Lambda

Ketika pengguna tidak ada di kolam pengguna saat masuk dengan kata sandi, atau dalam alur lupa kata sandi, Amazon Cognito memanggil pemicu ini. Setelah fungsi Lambda berhasil dikembalikan, Amazon Cognito membuat pengguna di kolam pengguna. Untuk detail tentang alur autentikasi dengan pemicu Lambda migrasi pengguna, lihat [Mengimpor pengguna ke kumpulan pengguna dengan migrasi pengguna pemicu Lambda](#)

Untuk memigrasi pemicu Lambda ini untuk memigrasi pengguna dari direktori pengguna yang ada ke kolam pengguna Amazon Cognito saat masuk, atau selama alur lupa kata sandi, gunakan pemicu Lambda ini.

### Topik

- [Memigrasi sumber pemicu Lambda](#)
- [Memigrasi parameter pemicu Lambda](#)

- [Contoh: Memigrasi pengguna dengan kata sandi yang ada](#)

## Memigrasi sumber pemicu Lambda

Nilai triggerSource	Peristiwa
UserMigration_Authentication	Migrasi pengguna saat login.
UserMigration_ForgotPassword	Migrasi pengguna selama alur lupa kata sandi.

## Memigrasi parameter pemicu Lambda

Permintaan yang diteruskan Amazon Cognito ke fungsi Lambda ini adalah kombinasi dari parameter di bawah ini dan [parameter umum](#) yang ditambahkan Amazon Cognito ke semua permintaan.

### JSON

```
{
  "userName": "string",
  "request": {
    "password": "string",
    "validationData": {
      "string": "string",
      . . .
    },
    "clientMetadata": {
      "string": "string",
      . . .
    }
  },
  "response": {
    "userAttributes": {
      "string": "string",
      . . .
    },
    "finalUserStatus": "string",
    "messageAction": "string",
    "desiredDeliveryMediums": [ "string", . . . ],
    "forceAliasCreation": boolean,
    "enableSMSMFA": boolean
  }
}
```

```
}  
}
```

## Memigrasi parameter permintaan pengguna

### userName

Nama pengguna yang dimasukkan pengguna saat login.

### kata sandi

Kata sandi yang dimasukkan oleh pengguna saat masuk. Amazon Cognito tidak mengirimkan nilai ini dalam permintaan yang diprakarsai oleh alur lupa kata sandi.

### validationData

Satu atau lebih pasangan nilai-kunci yang berisi data validasi dalam permintaan masuk pengguna. Untuk meneruskan data ini ke fungsi Lambda Anda, Anda dapat menggunakan ClientMetadata parameter dalam tindakan [InitiateAuth](#) dan [AdminInitiateAuth](#) API.

### clientMetadata

Satu atau lebih pasangan nilai-kunci yang dapat Anda berikan sebagai masukan kustom ke fungsi Lambda untuk pemacu migrasi pemacu pengguna yang memigrasi pemacu pengguna yang migrasi. Untuk meneruskan data ini ke fungsi Lambda Anda, Anda dapat menggunakan ClientMetadata parameter dalam tindakan [AdminRespondToAuthChallenge](#) dan [ForgotPassword](#) API.

## Memigrasi parameter respons pengguna

### userAttributes


Bidang ini wajib diisi.

Bidang ini harus berisi satu atau lebih pasangan nilai-nama yang disimpan Amazon Cognito di profil pengguna di kolam pengguna Anda dan gunakan sebagai atribut pengguna. Anda dapat menyertakan atribut pengguna standar dan kustom. Atribut kustom memerlukan prefiks custom: untuk membedakannya dari atribut standar. Untuk informasi selengkapnya, lihat [Atribut kustom](#).

#### Note

Untuk mengatur ulang kata sandi mereka dalam alur lupa kata sandi, pengguna harus memiliki alamat email terverifikasi atau nomor telepon terverifikasi. Amazon Cognito

mengirimkan pesan yang berisi kode atur ulang kata sandi ke alamat email atau nomor telepon di atribut pengguna.

Atribut	Persyaratan
Atribut apa pun yang ditandai sebagai diperlukan saat Anda membuat kolom pengguna	Jika ada atribut diperlukan yang hilang selama migrasi, Amazon Cognito menggunakan nilai default.
username	<p>Diperlukan jika Anda mengonfigurasi kolom pengguna Anda dengan atribut alias selain nama pengguna untuk masuk, dan pengguna telah memasukkan nilai alias yang valid sebagai nama pengguna. Nilai alias ini bisa berupa alamat email, nama pengguna pilihan, atau nomor telepon.</p> <p>Jika permintaan dan kumpulan pengguna memenuhi persyaratan alias, respons dari fungsi Anda harus menetapkan <code>username</code> parameter yang diterimanya ke atribut alias. Juga, respons harus menetapkan nilai Anda sendiri ke atribut. <code>username</code> Jika pangkalan pengguna Anda tidak memenuhi ketentuan yang diperlukan untuk menetapkan yang diterima <code>username</code> ke alias, maka <code>username</code> parameter dalam respons harus sama persis dengan permintaan, atau dihilangkan.</p> <div data-bbox="553 1409 1507 1577" style="border: 1px solid #add8e6; border-radius: 10px; padding: 10px; margin-top: 10px;"> <p> <b>Note</b> username harus unik di kolom pengguna.</p> </div>

## finalUserStatus

Anda dapat mengatur parameter ini `CONFIRMED` untuk mengonfirmasi otomatis pengguna sehingga mereka dapat masuk dengan kata sandi sebelumnya. Ketika Anda menetapkan

penggunaCONFIRMED, mereka tidak perlu mengambil tindakan tambahan sebelum mereka dapat masuk. Jika Anda tidak mengatur atribut iniCONFIRMED, itu diatur keRESET\_REQUIRED.

Sebuah finalUserStatus RESET\_REQUIRED cara bahwa pengguna harus mengubah kata sandi mereka segera setelah migrasi saat masuk, dan aplikasi klien Anda harus menangani PasswordResetRequiredException selama alur autentikasi.

#### Note

Amazon Cognito tidak menerapkan kebijakan kekuatan kata sandi yang Anda konfigurasi untuk pangkalan pengguna selama migrasi menggunakan pemicu Lambda. Jika kata sandi tidak memenuhi kebijakan kata sandi yang Anda konfigurasi, Amazon Cognito tetap menerima kata sandi sehingga dapat terus memigrasi pengguna. Untuk memberlakukan kebijakan kekuatan kata sandi dan menolak kata sandi yang tidak memenuhi kebijakan, validasi kekuatan kata sandi dalam kode Anda. Kemudian, jika kata sandi tidak memenuhi kebijakan, atur finalUserStatus keRESET\_REQUIRED.

#### messageAction

Anda dapat mengatur parameter ini untuk menolak SUPPRESS untuk mengirim pesan selamat datang yang biasanya dikirim Amazon Cognito ke pengguna baru. Jika fungsi Anda tidak mengembalikan parameter ini, Amazon Cognito akan mengirimkan pesan selamat datang.

#### desiredDeliveryMediums

Anda dapat mengatur parameter ini EMAIL untuk mengirim pesan selamat datang melalui email, atau SMS untuk mengirim pesan selamat datang melalui SMS. Jika fungsi Anda tidak mengembalikan parameter ini, Amazon Cognito mengirimkan pesan selamat datang melalui SMS.

#### forceAliasCreation

Jika Anda mengatur parameter ini TRUE dan nomor telepon atau alamat email dalam UserAttributes parameter sudah ada sebagai alias dengan pengguna yang berbeda, panggilan API memigrasi alias dari pengguna sebelumnya ke pengguna yang baru dibuat. Pengguna sebelumnya tidak akan lagi dapat masuk menggunakan alias tersebut.

Jika Anda menyetel parameter ini FALSE dan alias ada, Amazon Cognito tidak akan memigrasi pengguna dan mengembalikan kesalahan ke aplikasi klien.

Jika Anda tidak mengembalikan parameter ini, Amazon Cognito menganggap nilainya "false".



## AktifkanSMSMFA

Memigrasi parameter ini `true` untuk meminta pengguna yang diigrasi untuk masuk. Pool pengguna Anda harus mengaktifkan MFA. Atribut pengguna Anda dalam parameter permintaan harus menyertakan nomor telepon, atau migrasi pengguna tersebut akan gagal.

### Contoh: Memigrasi pengguna dengan kata sandi yang ada

Contoh fungsi Lambda ini memigrasikan pengguna dengan kata sandi yang ada dan menekan pesan selamat datang dari Amazon Cognito.

#### Node.js

```
const validUsers = {
  belladonna: { password: "Test123", emailAddress: "bella@example.com" },
};

// Replace this mock with a call to a real authentication service.
const authenticateUser = (username, password) => {
  if (validUsers[username] && validUsers[username].password === password) {
    return validUsers[username];
  } else {
    return null;
  }
};

const lookupUser = (username) => {
  const user = validUsers[username];

  if (user) {
    return { emailAddress: user.emailAddress };
  } else {
    return null;
  }
};

const handler = async (event) => {
  if (event.triggerSource == "UserMigration_Authentication") {
    // Authenticate the user with your existing user directory service
    const user = authenticateUser(event.userName, event.request.password);
    if (user) {
      event.response.userAttributes = {
```

```
        email: user.emailAddress,
        email_verified: "true",
    };
    event.response.finalUserStatus = "CONFIRMED";
    event.response.messageAction = "SUPPRESS";
}
} else if (event.triggerSource == "UserMigration_ForgotPassword") {
    // Look up the user in your existing user directory service
    const user = lookupUser(event.userName);
    if (user) {
        event.response.userAttributes = {
            email: user.emailAddress,
            // Required to enable password-reset code to be sent to user
            email_verified: "true",
        };
        event.response.messageAction = "SUPPRESS";
    }
}

return event;
};

export { handler };
```

## Pesan khusus Lambda pemicu

Amazon Cognito memanggil pemicu ini sebelum mengirim email atau pesan verifikasi telepon atau kode otentikasi multi-faktor (MFA). Anda dapat menyesuaikan pesan secara dinamis dengan pemicu pesan khusus Anda. Anda dapat mengedit pesan kustom statis di tab Penyesuaian pesan di konsol [Amazon Cognito](#) asli.

Permintaan termasuk `codeParameter`. Ini adalah string yang bertindak sebagai pengganti untuk kode yang diberikan Amazon Cognito kepada pengguna. Masukkan `codeParameter` string ke badan pesan tempat Anda ingin kode verifikasi muncul. Saat Amazon Cognito menerima respons ini, Amazon Cognito mengganti string dengan kode verifikasi `codeParameter` yang sebenarnya.

### Note

Fungsi Lambda pesan khusus dengan sumber `CustomMessage_AdminCreateUser` pemicu mengembalikan nama pengguna dan kode verifikasi. Karena pengguna yang dibuat

admin harus menerima nama pengguna dan kode mereka, respons dari fungsi Anda harus menyertakan keduanya dan `request.usernameParameter` `request.codeParameter`

## Topik

- [Pesan khusus Lambda memicu sumber](#)
- [Parameter pemicu Lambda pesan khusus](#)
- [Pesan khusus untuk contoh pendaftaran](#)
- [Pesan kustom untuk admin membuat contoh pengguna](#)

## Pesan khusus Lambda memicu sumber

Nilai triggerSource	Peristiwa
CustomMessage_SignUp	Pesan kustom – Untuk mengirim kode konfirmasi pasca pendaftaran.
CustomMessage_AdminCreateUser	Pesan kustom – Untuk mengirim kata sandi sementara ke pengguna baru.
CustomMessage_ResendCode	Pesan kustom – Untuk mengirim ulang kode konfirmasi ke pengguna yang sudah ada.
CustomMessage_ForgotPassword	Pesan kustom – Untuk mengirim kode konfirmasi untuk permintaan Lupa Kata Sandi.
CustomMessage_UpdateUserAttribute	Pesan kustom – Ketika email atau nomor telepon pengguna diubah, pemicu ini mengirimkan kode verifikasi secara otomatis kepada pengguna. Tidak dapat digunakan untuk atribut lainnya.
CustomMessage_VerifyUserAttribute	Pesan kustom – Pemicu ini mengirimkan kode verifikasi kepada pengguna ketika mereka memintanya secara manual untuk email atau nomor telepon baru.

Nilai triggerSource	Peristiwa
CustomMessage_Authentication	Pesan kustom – Untuk mengirim kode otentikasi i multifaktor (MFA) selama autentikasi.

## Parameter pemacu Lambda pesan khusus

Permintaan yang diteruskan Amazon Cognito ke fungsi Lambda ini adalah kombinasi dari parameter di bawah ini dan parameter [umum yang ditambahkan Amazon Cognito](#) ke semua permintaan.

### JSON

```
{
  "request": {
    "userAttributes": {
      "string": "string",
      . . .
    }
    "codeParameter": "####",
    "usernameParameter": "string",
    "clientMetadata": {
      "string": "string",
      . . .
    }
  },
  "response": {
    "smsMessage": "string",
    "emailMessage": "string",
    "emailSubject": "string"
  }
}
```

## Parameter permintaan pesan khusus

### userAttributes

Satu atau lebih pasangan nilai-nama yang mewakili atribut pengguna.

### codeParameter

String untuk Anda gunakan sebagai pengganti kode verifikasi dalam pesan kustom.

## usernameParameter

Nama pengguna. Amazon Cognito menyertakan parameter ini dalam permintaan yang dihasilkan dari pengguna yang dibuat admin.

## clientMetadata

Satu atau lebih pasangan nilai-kunci yang dapat Anda berikan sebagai masukan kustom ke fungsi Lambda yang Anda tentukan untuk pemicu pesan kustom. Permintaan yang memanggil fungsi pesan khusus tidak menyertakan data yang diteruskan dalam ClientMetadata parameter [AdminInitiateAuth](#) dan operasi [InitiateAuth](#) API. Untuk meneruskan data ini ke fungsi Lambda, Anda dapat menggunakan ClientMetadata parameter dalam tindakan API berikut:

- [AdminResetUserPassword](#)
- [AdminRespondToAuthChallenge](#)
- [AdminUpdateUserAttributes](#)
- [ForgotPassword](#)
- [GetUserAttributeVerificationCode](#)
- [ResendConfirmationCode](#)
- [SignUp](#)
- [UpdateUserAttributes](#)

## Parameter respons pesan khusus

Dalam respons, tentukan teks kustom yang akan digunakan dalam pesan kepada pengguna Anda. Untuk batasan string yang diterapkan Amazon Cognito pada parameter ini, lihat.

### [MessageTemplateType](#)

## smsMessage

Pesan SMS kustom yang akan dikirim ke pengguna Anda. Harus menyertakan `codeParameter` nilai yang Anda terima dalam permintaan.

## emailMessage

Pesan email khusus untuk dikirim ke pengguna Anda. Anda dapat menggunakan pemformatan HTML dalam `emailMessage` parameter. Harus menyertakan `codeParameter` nilai yang Anda terima dalam permintaan sebagai variabel `{####}`. Amazon Cognito dapat menggunakan `emailMessage` parameter hanya jika

EmailSendingAccount atribut kumpulan pengguna adalah DEVELOPER. Jika EmailSendingAccount atribut kumpulan pengguna tidak DEVELOPER dan emailMessage parameter dikembalikan, Amazon Cognito menghasilkan kode kesalahan 400. `com.amazonaws.cognito.identity.idp.model.InvalidLambdaResponseException`. Saat Anda memilih Amazon Simple Email Service (Amazon SES) untuk mengirim pesan email, EmailSendingAccount atribut kumpulan DEVELOPER pengguna adalah COGNITO\_DEFAULT.

### emailSubject

Baris subjek untuk pesan kustom. Anda hanya dapat menggunakan emailSubject parameter jika EmailSendingAccount atribut kumpulan pengguna adalah DEVELOPER. Jika EmailSendingAccount atribut kumpulan pengguna tidak DEVELOPER dan Amazon Cognito mengembalikan emailSubject parameter, Amazon Cognito akan menghasilkan kode kesalahan 400. `com.amazonaws.cognito.identity.idp.model.InvalidLambdaResponseException`. EmailSendingAccountAtribut kumpulan pengguna adalah DEVELOPER ketika Anda memilih untuk menggunakan Amazon Simple Email Service (Amazon SES) untuk mengirim pesan email. Kalau tidak, nilainya adalah COGNITO\_DEFAULT.

## Pesan khusus untuk contoh pendaftaran

Contoh fungsi Lambda ini menyesuaikan pesan email atau SMS saat layanan memerlukan aplikasi untuk mengirim kode verifikasi kepada pengguna.

Amazon Cognito dapat memanggil pemicu Lambda di beberapa acara: pasca-pendaftaran, mengirim ulang kode verifikasi, memulihkan kata sandi yang terlupakan, atau memverifikasi atribut pengguna. Respons mencakup pesan untuk SMS dan email. Pesan harus menyertakan parameter kode "####". Parameter ini adalah placeholder untuk kode verifikasi yang diterima pengguna.

Panjang maksimum untuk pesan email adalah 20.000 karakter UTF-8. Panjang ini termasuk kode verifikasi. Anda dapat menggunakan tag HTML dalam pesan email ini.

Panjang maksimum pesan SMS adalah 140 karakter UTF-8. Panjang ini termasuk kode verifikasi.

### Node.js

```
const handler = async (event) => {
  if (event.triggerSource === "CustomMessage_SignUp") {
```

```
const message = `Thank you for signing up. Your confirmation code is
${event.request.codeParameter}`;
event.response.smsMessage = message;
event.response.emailMessage = message;
event.response.emailSubject = "Welcome to the service.";
}
return event;
};

export { handler };
```

Amazon Cognito meneruskan informasi peristiwa ke fungsi Lambda Anda. Fungsi kemudian mengembalikan objek acara yang sama ke Amazon Cognito, dengan perubahan apa pun dalam respons. Di konsol Lambda, Anda dapat mengatur peristiwa pengujian dengan data yang relevan dengan pemicu Lambda Anda. Berikut ini adalah peristiwa pengujian untuk sampel kode ini:

## JSON

```
{
  "version": 1,
  "triggerSource": "CustomMessage_SignUp/CustomMessage_ResendCode/
CustomMessage_ForgotPassword/CustomMessage_VerifyUserAttribute",
  "region": "<region>",
  "userPoolId": "<userPoolId>",
  "userName": "<userName>",
  "callerContext": {
    "awsSdk": "<calling aws sdk with version>",
    "clientId": "<apps client id>",
    ...
  },
  "request": {
    "userAttributes": {
      "phone_number_verified": false,
      "email_verified": true,
      ...
    },
    "codeParameter": "####"
  },
  "response": {
    "smsMessage": "<custom message to be sent in the message with code parameter>"
    "emailMessage": "<custom message to be sent in the message with code
parameter>"
  }
}
```

```
    "emailSubject": "<custom email subject>"
  }
}
```

## Pesan kustom untuk admin membuat contoh pengguna

Permintaan yang dikirimkan Amazon Cognito ke contoh pesan khusus ini fungsi Lambda memiliki `triggerSource` nilai dan nama pengguna `CustomMessage_AdminCreateUser` dan kata sandi sementara. Fungsi ini mengisi `${event.request.codeParameter}` dari kata sandi sementara dalam permintaan, dan `${event.request.usernameParameter}` dari nama pengguna dalam permintaan.

Pesan kustom Anda harus menyisipkan nilai `codeParameter smsMessage` dan `usernameParameter` ke `emailMessage` dalam dan di objek respons. Dalam contoh ini, fungsi menulis pesan yang sama ke bidang respons `event.response.smsMessage` dan `event.response.emailMessage`.

Panjang maksimum pesan email adalah 20.000 karakter UTF-8. Panjang ini termasuk kode verifikasi. Anda dapat menggunakan tag HTML di email ini. Panjang maksimum pesan SMS adalah 140 karakter UTF-8. Panjang ini termasuk kode verifikasi.

Respons mencakup pesan untuk SMS dan email.

### Node.js

```
const handler = async (event) => {
  if (event.triggerSource === "CustomMessage_AdminCreateUser") {
    const message = `Welcome to the service. Your user name is
    ${event.request.usernameParameter}. Your temporary password is
    ${event.request.codeParameter}`;
    event.response.smsMessage = message;
    event.response.emailMessage = message;
    event.response.emailSubject = "Welcome to the service";
  }
  return event;
};

export { handler }
```



Amazon Cognito meneruskan informasi peristiwa ke fungsi Lambda Anda. Fungsi kemudian mengembalikan objek acara yang sama ke Amazon Cognito, dengan perubahan apa pun dalam respons. Di konsol Lambda, Anda dapat mengatur peristiwa pengujian dengan data yang relevan dengan pemicu Lambda Anda. Berikut ini adalah peristiwa pengujian untuk sampel kode ini:

## JSON

```
{
  "version": 1,
  "triggerSource": "CustomMessage_AdminCreateUser",
  "region": "<region>",
  "userPoolId": "<userPoolId>",
  "userName": "<userName>",
  "callerContext": {
    "awsSdk": "<calling aws sdk with version>",
    "clientId": "<apps client id>",
    ...
  },
  "request": {
    "userAttributes": {
      "phone_number_verified": false,
      "email_verified": true,
      ...
    },
    "codeParameter": "####",
    "usernameParameter": "username"
  },
  "response": {
    "smsMessage": "<custom message to be sent in the message with code parameter and username parameter>"
    "emailMessage": "<custom message to be sent in the message with code parameter and username parameter>"
    "emailSubject": "<custom email subject>"
  }
}
```

## Pemicu Lambda pengirim kustom

Kolam pengguna Amazon Cognito menyediakan pemicu `LambdaCustomEmailSender` dan `CustomSMSSender` untuk mengaktifkan pemberitahuan email dan SMS pihak ketiga. Anda dapat memilih penyedia SMS dan email untuk mengirim notifikasi kepada

pengguna dari dalam kode fungsi Lambda Anda. Ketika Amazon Cognito harus mengirim notifikasi seperti kode konfirmasi, kode verifikasi, atau kata sandi sementara kepada pengguna, peristiwa mengaktifkan fungsi Lambda yang dikonfigurasi. Amazon Cognito mengirimkan kode dan kata sandi sementara (rahasia) ke fungsi Lambda Anda yang diaktifkan. Amazon Cognito mengenkripsi rahasia ini dengan AWS KMS kunci dikelola pelanggan dan AWS Encryption SDK. AWS Encryption SDK adalah pustaka enkripsi di sisi klien yang membantu Anda untuk mengenkripsi dan mendekripsi data generik.

### Note

Untuk mengonfigurasi kolam pengguna Anda untuk menggunakan pemacu Lambda ini, Anda dapat menggunakan AWS CLI atau SDK. Konfigurasi ini tidak tersedia dari konsol Amazon Cognito.

## [CustomEmailSender](#)

Amazon Cognito memanggil pemacu ini untuk mengirim notifikasi email kepada pengguna.

## [CustomSMSSender](#)

Amazon Cognito memanggil pemacu ini untuk mengirim notifikasi SMS kepada pengguna.

## Sumber daya

Sumber daya berikut dapat membantu Anda untuk menggunakan `CustomEmailSender` dan `CustomSMSSender` Pemacu.

### AWS KMS

AWS KMS adalah layanan terkelola untuk membuat dan mengontrol AWS KMS kunci. Kunci ini mengenkripsi data Anda. Untuk informasi lebih lanjut, lihat, [Apa yang dimaksud dengan AWS Key Management Service?](#)

### Kunci KMS

Kunci KMS adalah representasi logis dari kunci kriptografi. Kunci KMS mencakup metadata, seperti ID kunci, tanggal pembuatan, deskripsi, dan status kunci. Kunci KMS juga terdiri dari bahan utama yang digunakan untuk mengenkripsi dan mendekripsi data. Untuk informasi selengkapnya, lihat, [AWS Kunci KMS](#).

## Kunci KMS simetris

Kunci KMS simetris adalah kunci enkripsi 256-bit yang tidak keluarAWS KMStidak terenkripsi. Untuk menggunakan kunci KMS simetris, Anda harus memanggilAWS KMS. Amazon Cognito menggunakan kunci simetris. Kunci yang sama mengenkripsi dan mendekripsi. Untuk informasi selengkapnya, lihat,[Kunci simetris KMS](#).

## Pemicu Lambda pengirim email kustom

Saat Anda menetapkan pemicu pengirim email khusus ke kumpulan pengguna Anda, Amazon Cognito akan memanggil fungsi Lambda alih-alih perilaku defaultnya saat peristiwa pengguna mengharuskannya mengirim pesan email. Dengan pemicu pengirim khusus,AWS Lambdafungsi dapat mengirim pemberitahuan email ke pengguna Anda melalui metode dan penyedia yang Anda pilih. Kode kustom fungsi Anda harus memproses dan mengirimkan semua pesan email dari kumpulan pengguna Anda.

### Note

Saat ini, Anda tidak dapat menetapkan pemicu pengirim khusus di konsol Amazon Cognito. Anda dapat menetapkan pemicu denganLambdaConfigparameter dalamCreateUserPoolatauUpdateUserPoolPermintaan API.

Untuk mengatur pemicu ini, lakukan langkah-langkah berikut:

1. Buat[kunci enkripsi simetris](#)diAWS Key Management Service(AWS KMS). Amazon Cognito menghasilkan rahasia — kata sandi sementara, kode verifikasi, dan kode konfirmasi — kemudian menggunakan kunci KMS ini untuk mengenkripsi rahasia. Anda kemudian dapat menggunakan[Dekripsi](#)Operasi API dalam fungsi Lambda Anda untuk mendekripsi rahasia dan mengirimkannya ke pengguna dalam teks biasa. The[AWS Encryption SDK](#)adalah alat yang berguna untukAWS KMSoperasi dalam fungsi Anda.
2. Buat fungsi Lambda yang ingin Anda tetapkan sebagai pemicu pengirim kustom Anda. Hibahkms :Decrytizin untuk kunci KMS Anda ke peran fungsi Lambda.
3. Berikan kepala layanan Amazon Cognitioncognito-idp.amazonaws.comakses untuk memanggil fungsi Lambda.
4. Tulis kode fungsi Lambda yang mengarahkan pesan Anda ke metode pengiriman khusus atau penyedia pihak ketiga. Untuk mengirimkan verifikasi atau kode konfirmasi pengguna Anda, Base64

mendekode dan mendekripsi nilai `codeparameter` dalam permintaan. Operasi ini menghasilkan kode teks biasa atau kata sandi yang harus Anda sertakan dalam pesan Anda.

- Perbarui kumpulan pengguna sehingga menggunakan pemicu Lambda pengirim khusus. Prinsipal IAM yang memperbarui atau membuat kumpulan pengguna dengan pemicu pengirim khusus harus memiliki izin untuk membuat hibah untuk kunci KMS Anda. Berikut ini `LambdaConfig` cuplikan menetapkan fungsi pengirim SMS dan email kustom.

```
"LambdaConfig": {
  "KMSKeyID": "arn:aws:kms:us-
east-1:123456789012:key/a6c4f8e2-0c45-47db-925f-87854bc9e357",
  "CustomEmailSender": {
    "LambdaArn": "arn:aws:lambda:us-east-1:123456789012:function:MyFunction",
    "LambdaVersion": "V1_0"
  },
  "CustomSMSSender": {
    "LambdaArn": "arn:aws:lambda:us-east-1:123456789012:function:MyFunction",
    "LambdaVersion": "V1_0"
  }
}
```

### Parameter pemicu Lambda pengirim email kustom

Permintaan yang diteruskan Amazon Cognito ke fungsi Lambda ini adalah kombinasi dari parameter di bawah ini dan [parameter umum](#) yang ditambahkan Amazon Cognito ke semua permintaan.

### JSON

```
{
  "request": {
    "type": "customEmailSenderRequestV1",
    "code": "string",
    "clientMetadata": {
      "string": "string",
      . . .
    },
    "userAttributes": {
      "string": "string",
      . . .
    }
  }
}
```

## Parameter permintaan pengirim email kustom

### jenis

Versi permintaan. Untuk acara pengirim email kustom, nilai string ini selalucustomEmailSenderRequestV1.

### code

Kode terenkripsi yang fungsi Anda dapat mendekripsi dan mengirim ke pengguna Anda.

### clientMetadata

Satu atau beberapa pasangan nilai kunci yang dapat Anda berikan sebagai input khusus ke pemacu fungsi Lambda pengirim email kustom. Untuk meneruskan data ini ke fungsi Lambda Anda, Anda dapat menggunakan `ClientMetadataParameter` di [AdminRespondToAuthChallenge](#) dan [RespondToAuthChallenge](#) Tindakan API. Amazon Cognito tidak menyertakan data dari `ClientMetadataParameter` dalam [AdminInitiateAuth](#) dan [InitiateAuth](#) Operasi API dalam permintaan yang diteruskan ke fungsi otentikasi pos.

### userAttributes

Satu atau lebih pasangan kunci-nilai yang mewakili atribut pengguna.

## Parameter respons pengirim email kustom

Amazon Cognito tidak mengharapkan informasi pengembalian tambahan dalam respons pengirim email khusus. Fungsi Anda dapat menggunakan operasi API untuk menanyakan dan memodifikasi sumber daya Anda, atau merekam metadata peristiwa ke sistem eksternal.

## Mengaktifkan pemacu Lambda pengirim email kustom

Untuk mengatur pemacu pengirim email kustom yang menggunakan logika kustom untuk mengirim pesan email ke kumpulan pengguna Anda, aktifkan pemacu sebagai berikut. Prosedur berikut ini menetapkan pemacu email khusus, pemacu SMS khusus, atau keduanya ke kumpulan pengguna Anda. Setelah Anda menambahkan pemacu pengirim email khusus Anda, Amazon Cognito selalu mengirimkan atribut pengguna, termasuk alamat email, dan kode satu kali ke fungsi Lambda Anda ketika seharusnya mengirim pesan email dengan Amazon Simple Email Service.

**⚠ Important**

Amazon Cognito HTML-lolos dari karakter yang dicadangkan seperti `<(&lt;)` dan `>(&gt;)` di kata sandi sementara pengguna Anda. Karakter ini mungkin muncul dalam kata sandi sementara yang dikirimkan Amazon Cognito ke fungsi pengirim email khusus Anda, tetapi tidak muncul dalam kode verifikasi sementara. Untuk mengirim kata sandi sementara, fungsi Lambda Anda harus melepaskan karakter ini setelah mendekripsi kata sandi, dan sebelum mengirim pesan ke pengguna Anda.

1. Buat sebuah kunci enkripsi di AWS KMS. Kunci ini mengenkripsi kata sandi sementara dan kode otorisasi yang dihasilkan Amazon Cognito. Anda kemudian dapat mendekripsi rahasia ini dalam fungsi Lambda pengirim khusus dan mengirimkannya ke pengguna Anda dalam teks biasa.
2. Berikan kepala layanan Amazon Cognito `cognito-idp.amazonaws.com` akses untuk mengenkripsi kode dengan kunci KMS.

Terapkan kebijakan berbasis sumber daya berikut ke kunci KMS Anda.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Principal": {
      "Service": "cognito-idp.amazonaws.com"
    },
    "Action": "kms:CreateGrant",
    "Resource": "arn:aws:kms:us-  
west-2:111222333444:key/1example-2222-3333-4444-999example",
    "Condition": {
      "StringEquals": {
        "aws:SourceAccount": "111222333444"
      },
      "ArnLike": {
        "aws:SourceArn": "arn:aws:cognito-idp:us-  
west-2:111222333444:userpool/us-east-1_EXAMPLE"
      }
    }
  }]
}
```

3. Buat fungsi Lambda untuk pemicu pengirim kustom. Amazon Cognito menggunakan [AWSenkripsi SDK](#) untuk mengenkripsi rahasia, kata sandi sementara, dan kode yang mengotorisasi permintaan API pengguna Anda.
  - Tetapkan peran IAM ke fungsi Lambda Anda yang memiliki, setidaknya, `kms:Decrypt` izin untuk kunci KMS Anda.
4. Berikan kepala layanan Amazon Cognito `cognito-idp.amazonaws.com` akses untuk memanggil fungsi Lambda.

Berikut ini AWS CLI perintah memberikan izin Amazon Cognito untuk menjalankan fungsi Lambda Anda:

```
aws lambda add-permission --function-name lambda_arn --statement-id  
"CognitoLambdaInvokeAccess" --action lambda:InvokeFunction --principal cognito-  
idp.amazonaws.com
```

5. Tulis kode fungsi Lambda Anda untuk mengirim pesan Anda. Amazon Cognito menggunakan AWS Encryption SDK untuk mengenkripsi rahasia sebelum Amazon Cognito mengirimkan rahasia ke fungsi Lambda pengirim khusus. Dalam fungsi Anda, dekripsi rahasia dan proses metadata yang relevan. Kemudian kirim kode, pesan kustom Anda sendiri, dan nomor telepon tujuan ke API kustom yang mengirimkan pesan Anda.
6. Tambahkan AWS Encryption SDK untuk fungsi Lambda Anda. Untuk informasi lebih lanjut, lihat [AWS Enkripsi bahasa pemrograman SDK](#). Untuk memperbarui paket Lambda, selesaikan langkah-langkah berikut.
  - a. Ekspor fungsi Lambda Anda sebagai file.zip di AWS Management Console.
  - b. Buka fungsi Anda dan tambahkan AWS Encryption SDK. Untuk informasi selengkapnya dan tautan unduhan, lihat [AWS Encryption SDK bahasa pemrograman](#) di AWS Encryption SDK Panduan Pengembang.
  - c. Zip fungsi Anda dengan dependensi SDK Anda, dan unggah fungsi tersebut ke Lambda. Untuk informasi lebih lanjut, lihat [Menyebarkan fungsi Lambda sebagai arsip file.zip](#) di AWS Lambda Panduan Pengembang.
7. Perbarui kumpulan pengguna Anda untuk menambahkan pemicu Lambda pengirim kustom. Sertakan `CustomSMSSender` atau `CustomEmailSender` parameter dalam sebuah `UpdateUserPool` permintaan API. The `UpdateUserPool` operasi API membutuhkan semua parameter kumpulan pengguna Anda dan parameter yang ingin Anda ubah. Jika Anda

tidak memberikan semua parameter yang relevan, Amazon Cognito menetapkan nilai parameter yang hilang ke defaultnya. Seperti yang ditunjukkan dalam contoh berikut, sertakan entri untuk semua fungsi Lambda yang ingin Anda tambahkan atau simpan di kumpulan pengguna Anda. Untuk informasi selengkapnya, lihat [Memperbarui konfigurasi kumpulan pengguna](#).

```
#Send this parameter in an 'aws cognito-idp update-user-pool' CLI command,
including any existing
#user pool configurations.

--lambda-config "PreSignUp=lambda-arn, \
                CustomSMSSender={LambdaVersion=V1_0,LambdaArn=lambda-arn}, \
                CustomEmailSender={LambdaVersion=V1_0,LambdaArn=lambda-arn}, \
                KMSKeyID=key-id"
```

Untuk menghapus pemacu Lambda pengirim kustom dengan `update-user-pool` AWS CLI, hilangkan `CustomSMSSender` atau `CustomEmailSender` parameter dari `--lambda-config`, dan sertakan semua pemacu lain yang ingin Anda gunakan dengan kumpulan pengguna Anda.

Untuk menghapus pemacu Lambda pengirim kustom dengan `UpdateUserPool` Permintaan API, hilangkan `CustomSMSSender` atau `CustomEmailSender` parameter dari badan permintaan yang berisi sisa konfigurasi kumpulan pengguna Anda.

## Contoh kode

Contoh Node.js berikut memproses peristiwa pesan email dalam fungsi Lambda pengirim email kustom Anda. Contoh ini mengasumsikan fungsi Anda memiliki dua variabel lingkungan yang ditentukan.

### KEY\_ALIAS

The [alias](#) kunci KMS yang ingin Anda gunakan untuk mengenkripsi dan mendekripsi kode pengguna Anda.

### KEY\_ARN

Nama Sumber Daya Amazon (ARN) dari kunci KMS yang ingin Anda gunakan untuk mengenkripsi dan mendekripsi kode pengguna Anda.



```

const AWS = require('aws-sdk');
const b64 = require('base64-js');
const encryptionSdk = require('@aws-crypto/client-node');
//Configure the encryption SDK client with the KMS key from the environment variables.
const { encrypt, decrypt } =
  encryptionSdk.buildClient(encryptionSdk.CommitmentPolicy.REQUIRE_ENCRYPT_ALLOW_DECRYPT);
const generatorKeyId = process.env.KEY_ALIAS;
const keyIds = [ process.env.KEY_ARN ];
const keyring = new encryptionSdk.KmsKeyringNode({ generatorKeyId, keyIds })
exports.handler = async (event) => {
  //Decrypt the secret code using encryption SDK.
  let plainTextCode;
  if(event.request.code){
    const { plaintext, messageHeader } = await decrypt(keyring,
b64.toByteArray(event.request.code));
    plainTextCode = plaintext
  }
  //PlainTextCode now contains the decrypted secret.
  if(event.triggerSource == 'CustomEmailSender_SignUp'){
    //Send an email message to your user via a custom provider.
    //Include the temporary password in the message.
  }
  else if(event.triggerSource == 'CustomEmailSender_ResendCode'){
  }
  else if(event.triggerSource == 'CustomEmailSender_ForgotPassword'){
  }
  else if(event.triggerSource == 'CustomEmailSender_UpdateUserAttribute'){
  }
  else if(event.triggerSource == 'CustomEmailSender_VerifyUserAttribute'){
  }
  else if(event.triggerSource == 'CustomEmailSender_AdminCreateUser'){
  }
  else if(event.triggerSource == 'CustomEmailSender_AccountTakeOverNotification'){
  }
  return;
};

```

## Sumber pemicu Lambda pengirim email khusus

Tabel berikut menunjukkan peristiwa pemicu untuk sumber pemicu email kustom dalam kode Lambda Anda.

TriggerSource value	Peristiwa
CustomEmailSender_SignUp	Seorang pengguna mendaftar dan Amazon Cognito mengirimkan pesan selamat datang.
CustomEmailSender_ForgotPassword	Seorang pengguna meminta kode untuk mengatur ulang kata sandi mereka.
CustomEmailSender_ResendCode	Pengguna meminta kode pengganti untuk mengatur ulang kata sandi mereka.
CustomEmailSender_UpdateUserAttribute	Pengguna memperbarui alamat email atau atribut nomor telepon dan Amazon Cognito mengirimkan kode untuk memverifikasi atribut.
CustomEmailSender_VerifyUserAttribute	Pengguna membuat atribut alamat email atau nomor telepon baru dan Amazon Cognito mengirimkan kode untuk memverifikasi atribut.
CustomEmailSender_AdminCreateUser	Anda membuat pengguna baru di kumpulan pengguna Anda dan Amazon Cognito mengiriminya mereka kata sandi sementara.
CustomEmailSender_AccountTakeOverNotification	Amazon Cognito mendeteksi upaya untuk mengambil alih akun pengguna dan mengirimkan pemberitahuan kepada pengguna.

## Pemicu Lambda pengirim SMS kustom

Saat Anda menetapkan pemicu pengirim SMS khusus ke kumpulan pengguna Anda, Amazon Cognito akan memanggil fungsi Lambda alih-alih perilaku defaultnya saat peristiwa pengguna mengharuskannya mengirim pesan SMS. Dengan pemicu pengirim khusus, AWS Lambda fungsi Anda dapat mengirim pemberitahuan SMS ke pengguna Anda melalui metode dan penyedia yang Anda pilih. Kode kustom fungsi Anda harus memproses dan mengirimkan semua pesan SMS dari kumpulan pengguna Anda.

**Note**

Saat ini, Anda tidak dapat menetapkan pemicu pengirim khusus di konsol Amazon Cognito. Anda dapat menetapkan pemicu dengan LambdaConfig parameter dalam permintaan CreateUserPool atau UpdateUserPool API.

Untuk mengatur pemicu ini, lakukan langkah-langkah berikut:

1. Buat [kunci enkripsi simetris](#) di AWS Key Management Service (AWS KMS). Amazon Cognito menghasilkan rahasia — kata sandi sementara, kode verifikasi, dan kode konfirmasi — kemudian menggunakan kunci KMS ini untuk mengenkripsi rahasia. Anda kemudian dapat menggunakan operasi [Decrypt](#) API di fungsi Lambda Anda untuk mendekripsi rahasia dan mengirimkannya ke pengguna dalam plaintext. [AWS Encryption SDK](#) ini adalah alat yang berguna untuk AWS KMS operasi dalam fungsi Anda.
2. Buat fungsi Lambda yang ingin Anda tetapkan sebagai pemicu pengirim kustom Anda. Berikan kms:Decrypt izin untuk kunci KMS Anda ke peran fungsi Lambda.
3. Berikan cognito-idp.amazonaws.com akses utama layanan Amazon Cognito untuk menjalankan fungsi Lambda.
4. Tulis kode fungsi Lambda yang mengarahkan pesan Anda ke metode pengiriman khusus atau penyedia pihak ketiga. Untuk mengirimkan verifikasi atau kode konfirmasi pengguna Anda, Base64 mendekode dan mendekripsi nilai code parameter dalam permintaan. Operasi ini menghasilkan kode teks biasa atau kata sandi yang harus Anda sertakan dalam pesan Anda.
5. Perbarui kumpulan pengguna sehingga menggunakan pemicu Lambda pengirim khusus. Prinsipal IAM yang memperbarui atau membuat kumpulan pengguna dengan pemicu pengirim khusus harus memiliki izin untuk membuat hibah untuk kunci KMS Anda. LambdaConfig cuplikan berikut menetapkan fungsi SMS dan pengirim email khusus.

```
"LambdaConfig": {
  "KMSKeyID": "arn:aws:kms:us-
east-1:123456789012:key/a6c4f8e2-0c45-47db-925f-87854bc9e357",
  "CustomEmailSender": {
    "LambdaArn": "arn:aws:lambda:us-east-1:123456789012:function:MyFunction",
    "LambdaVersion": "V1_0"
  },
  "CustomSMSSender": {
    "LambdaArn": "arn:aws:lambda:us-east-1:123456789012:function:MyFunction",
    "LambdaVersion": "V1_0"
  }
}
```

```
}
```

## Parameter pemicu Lambda pengirim SMS kustom

Permintaan yang diteruskan Amazon Cognito ke fungsi Lambda ini adalah kombinasi dari parameter di bawah ini dan parameter [umum yang ditambahkan Amazon](#) Cognito ke semua permintaan.

### JSON

```
{
  "request": {
    "type": "customSMSSenderRequestV1",
    "code": "string",
    "clientMetadata": {
      "string": "string",
      . . .
    },
    "userAttributes": {
      "string": "string",
      . . .
    }
  }
}
```

## Parameter permintaan pengirim SMS kustom

### tipe

Versi permintaan. Untuk acara pengirim SMS khusus, nilai string ini selalu `customSMSSenderRequestV1`.

### code

Kode terenkripsi yang fungsi Anda dapat mendekripsi dan mengirim ke pengguna Anda.

### clientMetadata

Satu atau lebih pasangan nilai kunci yang dapat Anda berikan sebagai input khusus ke pemicu fungsi Lambda pengirim SMS kustom. Untuk meneruskan data ini ke fungsi Lambda Anda, Anda dapat menggunakan ClientMetadata parameter dalam tindakan [AdminRespondToAuthChallenge](#) dan [RespondToAuthChallenge](#) API. Amazon Cognito tidak

menyertakan data dari ClientMetadata parameter dalam [AdminInitiateAuth](#) dan operasi [InitiateAuth](#) API dalam permintaan yang diteruskan ke fungsi otentikasi pos.

userAttributes

Satu atau lebih pasangan kunci-nilai yang mewakili atribut pengguna.

Parameter respons pengirim SMS kustom

Amazon Cognito tidak mengharapkan informasi pengembalian tambahan dalam tanggapan. Fungsi Anda dapat menggunakan operasi API untuk menanyakan dan memodifikasi sumber daya Anda, atau merekam metadata peristiwa ke sistem eksternal.

Mengaktifkan pemicu Lambda pengirim SMS kustom

Anda dapat mengatur pemicu pengirim SMS khusus yang menggunakan logika khusus untuk mengirim pesan SMS ke kumpulan pengguna Anda. Prosedur berikut menetapkan pemicu SMS khusus, pemicu email khusus, atau keduanya ke kumpulan pengguna Anda. Setelah menambahkan pemicu pengirim SMS khusus, Amazon Cognito selalu mengirimkan atribut pengguna, termasuk nomor telepon, dan kode satu kali ke fungsi Lambda Anda, bukan perilaku default yang mengirim pesan SMS dengan Amazon Simple Notification Service.

#### Important

Amazon Cognito HTML lolos dari karakter yang dicadangkan seperti `<` (`&lt;`) dan `>` (`&gt;`) di kata sandi sementara pengguna Anda. Karakter ini mungkin muncul dalam kata sandi sementara yang dikirimkan Amazon Cognito ke fungsi pengirim email khusus Anda, tetapi tidak muncul dalam kode verifikasi sementara. Untuk mengirim kata sandi sementara, fungsi Lambda Anda harus melepaskan karakter ini setelah mendekripsi kata sandi, dan sebelum mengirim pesan ke pengguna Anda.

1. Buat sebuah kunci enkripsi di AWS KMS. Kunci ini mengenkripsi kata sandi sementara dan kode otorisasi yang dihasilkan Amazon Cognito. Anda kemudian dapat mendekripsi rahasia ini dalam fungsi Lambda pengirim khusus dan mengirimkannya ke pengguna Anda dalam teks biasa.
2. Berikan `cognito-idp.amazonaws.com` akses utama layanan Amazon Cognito untuk mengenkripsi kode dengan kunci KMS.

Terapkan kebijakan berbasis sumber daya berikut ke kunci KMS Anda.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Principal": {
      "Service": "cognito-idp.amazonaws.com"
    },
    "Action": "kms:CreateGrant",
    "Resource": "arn:aws:kms:us-
west-2:111222333444:key/1example-2222-3333-4444-999example",
    "Condition": {
      "StringEquals": {
        "aws:SourceAccount": "111222333444"
      },
      "ArnLike": {
        "aws:SourceArn": "arn:aws:cognito-idp:us-
west-2:111222333444:userpool/us-east-1_EXAMPLE"
      }
    }
  }]
}
```

3. Buat fungsi Lambda untuk pemicu pengirim kustom. Amazon Cognito menggunakan [SDK AWS enkripsi](#) untuk mengenkripsi rahasia, kata sandi sementara, dan kode yang mengizinkan permintaan API pengguna Anda.
  - Tetapkan peran IAM ke fungsi Lambda Anda yang memiliki, setidaknya, `kms:Decrypt` izin untuk kunci KMS Anda.
4. Berikan `cognito-idp.amazonaws.com` akses utama layanan Amazon Cognito untuk menjalankan fungsi Lambda.

AWS CLI Perintah berikut memberikan izin Amazon Cognito untuk menjalankan fungsi Lambda Anda:

```
aws lambda add-permission --function-name lambda_arn --statement-id
"CognitoLambdaInvokeAccess" --action lambda:InvokeFunction --principal cognito-
idp.amazonaws.com
```

5. Tulis kode fungsi Lambda Anda untuk mengirim pesan Anda. Amazon Cognito menggunakan AWS Encryption SDK untuk mengenkripsi rahasia sebelum Amazon Cognito mengirimkan rahasia ke fungsi Lambda pengirim khusus. Dalam fungsi Anda, dekripsi rahasia dan proses metadata yang relevan. Kemudian kirim kode, pesan kustom Anda sendiri, dan nomor telepon tujuan ke API kustom yang mengirimkan pesan Anda.
6. Tambahkan AWS Encryption SDK ke fungsi Lambda Anda. Untuk informasi selengkapnya, lihat [AWS Enkripsi bahasa pemrograman SDK](#). Untuk memperbarui paket Lambda, selesaikan langkah-langkah berikut.
  - a. Ekspor fungsi Lambda Anda sebagai file.zip di file. AWS Management Console
  - b. Buka fungsi Anda dan tambahkan AWS Encryption SDK. Untuk informasi selengkapnya dan tautan unduhan, lihat [bahasa AWS Encryption SDK pemrograman](#) di Panduan AWS Encryption SDK Pengembang.
  - c. Zip fungsi Anda dengan dependensi SDK Anda, dan unggah fungsi tersebut ke Lambda. Untuk informasi selengkapnya, lihat [Menerapkan fungsi Lambda sebagai arsip file.zip](#) di AWS Lambda Panduan Pengembang.
7. Perbarui kumpulan pengguna Anda untuk menambahkan pemicu Lambda pengirim kustom. Sertakan CustomEmailSender parameter CustomSMSSender atau dalam permintaan UpdateUserPool API. Operasi UpdateUserPool API memerlukan semua parameter kumpulan pengguna Anda dan parameter yang ingin Anda ubah. Jika Anda tidak memberikan semua parameter yang relevan, Amazon Cognito menetapkan nilai parameter yang hilang ke defaultnya. Seperti yang ditunjukkan dalam contoh berikut, sertakan entri untuk semua fungsi Lambda yang ingin Anda tambahkan atau simpan di kumpulan pengguna Anda. Untuk informasi selengkapnya, lihat [Memperbarui konfigurasi kumpulan pengguna](#).

```
#Send this parameter in an 'aws cognito-idp update-user-pool' CLI command,
including any existing
#user pool configurations.

--lambda-config "PreSignUp=lambda-arn, \
                 CustomSMSSender={LambdaVersion=V1_0,LambdaArn=lambda-arn}, \
                 CustomEmailSender={LambdaVersion=V1_0,LambdaArn=lambda-arn},
\
                 KMSKeyID=key-id"
```

Untuk menghapus pemacu Lambda pengirim kustom dengan `update-user-pool` AWS CLI, hilangkan `CustomSMSSender` parameter `CustomEmailSender` atau `--lambda-config` dari, dan sertakan semua pemacu lain yang ingin Anda gunakan dengan kumpulan pengguna Anda.

Untuk menghapus pemacu Lambda pengirim kustom dengan `UpdateUserPool` permintaan API, hilangkan `CustomSMSSender` parameter `CustomEmailSender` atau dari badan permintaan yang berisi konfigurasi kumpulan pengguna lainnya.

### Contoh kode

Contoh Node.js berikut memproses peristiwa pesan SMS dalam fungsi Lambda pengirim SMS kustom Anda. Contoh ini mengasumsikan fungsi Anda memiliki dua variabel lingkungan yang ditentukan.

### KEY\_ALIAS

[Alias](#) kunci KMS yang ingin Anda gunakan untuk mengenkripsi dan mendekripsi kode pengguna Anda.

### KEY\_ARN

Nama Sumber Daya Amazon (ARN) dari kunci KMS yang ingin Anda gunakan untuk mengenkripsi dan mendekripsi kode pengguna Anda.

```
const AWS = require('aws-sdk');
const b64 = require('base64-js');
const encryptionSdk = require('@aws-crypto/client-node');
//Configure the encryption SDK client with the KMS key from the environment variables.

const { encrypt, decrypt } =
  encryptionSdk.buildClient(encryptionSdk.CommitmentPolicy.REQUIRE_ENCRYPT_ALLOW_DECRYPT);
const generatorKeyId = process.env.KEY_ALIAS;
const keyIds = [ process.env.KEY_ARN ];
const keyring = new encryptionSdk.KmsKeyringNode({ generatorKeyId, keyIds })
exports.handler = async (event) => {
  //Decrypt the secret code using encryption SDK.
  let plainTextCode;
  if(event.request.code){
    const { plaintext, messageHeader } = await decrypt(keyring,
      b64.toByteArray(event.request.code));
    plainTextCode = plaintext
  }
}
```



```
//PlainTextCode now contains the decrypted secret.
if(event.triggerSource == 'CustomSMSSender_SignUp'){
  //Send an SMS message to your user via a custom provider.
  //Include the temporary password in the message.
}
else if(event.triggerSource == 'CustomSMSSender_ResendCode'){
}
else if(event.triggerSource == 'CustomSMSSender_ForgotPassword'){
}
else if(event.triggerSource == 'CustomSMSSender_UpdateUserAttribute'){
}
else if(event.triggerSource == 'CustomSMSSender_VerifyUserAttribute'){
}
else if(event.triggerSource == 'CustomSMSSender_AdminCreateUser'){
}
else if(event.triggerSource == 'CustomSMSSender_AccountTakeOverNotification'){
}
return;
};
```

## Topik

- [Mengevaluasi kemampuan pesan SMS dengan fungsi pengirim SMS khusus](#)
- [Sumber pemicu Lambda pengirim SMS kustom](#)

## Mengevaluasi kemampuan pesan SMS dengan fungsi pengirim SMS khusus

Fungsi Lambda pengirim SMS khusus menerima pesan SMS yang akan dikirim oleh kumpulan pengguna Anda, dan fungsi tersebut mengirimkan konten berdasarkan logika kustom Anda. Amazon Cognito mengirimkan [Parameter pemicu Lambda pengirim SMS kustom](#) ke fungsi Anda. Fungsi Anda dapat melakukan apa yang Anda inginkan dengan informasi ini. Misalnya, Anda dapat mengirim kode ke topik Amazon Simple Notification Service (Amazon SNS). Pelanggan topik Amazon SNS dapat berupa pesan SMS, titik akhir HTTPS, atau alamat email.

[Untuk membuat lingkungan pengujian untuk perpesanan SMS Amazon Cognito dengan fungsi Lambda pengirim SMS khusus, amazon-cognito-user-poollihat development-and-testing-with - sms-redirected-to-email - di pustaka aws-samples pada. GitHub](#) Repositori berisi AWS CloudFormation template yang dapat membuat kumpulan pengguna baru, atau bekerja dengan kumpulan pengguna yang sudah Anda miliki. Template ini membuat fungsi Lambda dan topik Amazon SNS. Fungsi Lambda yang ditetapkan template sebagai pemicu pengirim SMS khusus, mengalihkan pesan SMS Anda ke pelanggan ke topik Amazon SNS.

Saat Anda menerapkan solusi ini ke kumpulan pengguna, semua pesan yang biasanya dikirim Amazon Cognito melalui pesan SMS, fungsi Lambda malah mengirim ke alamat email pusat. Gunakan solusi ini untuk menyesuaikan dan melihat pratinjau pesan SMS, dan untuk menguji peristiwa kumpulan pengguna yang menyebabkan Amazon Cognito mengirim pesan SMS. Setelah Anda menyelesaikan pengujian, putar kembali CloudFormation tumpukan, atau hapus tugas fungsi pengirim SMS khusus dari kumpulan pengguna Anda.

### Important

Jangan gunakan template di [amazon-cognito-user-pool- development-and-testing-with - sms-redirected-to-email](#) untuk membangun lingkungan produksi. Fungsi Lambda pengirim SMS khusus dalam solusi mensimulasikan pesan SMS, tetapi fungsi Lambda mengirimkan semuanya ke satu alamat email pusat. Sebelum Anda dapat mengirim pesan SMS dalam kumpulan pengguna Amazon Cognito produksi, Anda harus melengkapi persyaratan yang ditunjukkan di [Pengaturan pesan SMS untuk kolam pengguna Amazon Cognito](#)

Sumber pemicu Lambda pengirim SMS kustom

Tabel berikut menunjukkan peristiwa memicu untuk sumber pemicu SMS kustom dalam kode Lambda Anda.

TriggerSource value	Peristiwa
CustomSMSSender_SignUp	Seorang pengguna mendaftar dan Amazon Cognito mengirimkan pesan selamat datang.
CustomSMSSender_ForgotPassword	Pengguna meminta kode untuk mengatur ulang kata sandi mereka.
CustomSMSSender_ResendCode	Pengguna meminta kode baru untuk mengonfirmasi pendaftaran mereka.
CustomSMSSender_VerifyUserAttribute	Pengguna membuat atribut alamat email atau nomor telepon baru dan Amazon Cognito mengirimkan kode untuk memverifikasi atribut.

TriggerSource value	Peristiwa
CustomSMSSender_UpdateUserAttribute	Pengguna memperbarui alamat email atau atribut nomor telepon dan Amazon Cognito mengirimkan kode untuk memverifikasi atribut.
CustomSMSSender_Authentication	Pengguna yang dikonfigurasi dengan otentikasi multi-faktor SMS (MFA) masuk.
CustomSMSSender_AdminCreateUser	Anda membuat pengguna baru di kumpulan pengguna Anda dan Amazon Cognito mengiriminya kata sandi sementara.

## Menggunakan analitik Amazon Pinpoint dengan kumpulan pengguna Amazon Cognito

Kumpulan pengguna Amazon Cognito terintegrasi dengan Amazon Pinpoint untuk menyediakan analitik bagi kumpulan pengguna Amazon Cognito dan untuk memperkaya data pengguna untuk kampanye Amazon Pinpoint. Amazon Pinpoint menyediakan analitik dan kampanye yang ditargetkan untuk mendorong keterlibatan pengguna di aplikasi seluler menggunakan notifikasi push. Dengan dukungan analitik Amazon Pinpoint di kolam pengguna Amazon Cognito, Anda dapat melacak pendaftaran kolam pengguna, masuk ke kolam pengguna, autentikasi yang gagal, pengguna aktif harian (DAU), dan pengguna aktif bulanan (MAU) di konsol Amazon Pinpoint. Anda dapat menelusuri data untuk rentang tanggal atau atribut yang berbeda, seperti platform perangkat, lokal perangkat, dan versi aplikasi.

Anda juga dapat menyiapkan atribut khusus untuk aplikasi Anda. Itu kemudian dapat digunakan untuk menyegmentasikan pengguna Anda pada Amazon Pinpoint dan mengiriminya notifikasi push yang ditargetkan. Jika Anda memilih Bagikan data atribut pengguna dengan Amazon Pinpoint di tab Analytics di konsol Amazon Cognito, Amazon Pinpoint akan membuat titik akhir tambahan untuk alamat email pengguna dan nomor telepon.

Saat mengaktifkan analitik Amazon Pinpoint di kumpulan pengguna dengan konsol Amazon Cognito, Anda juga membuat [peran terkait layanan yang diasumsikan](#) oleh Amazon Cognito saat membuat permintaan API ke Amazon Pinpoint untuk kumpulan pengguna Anda. Prinsip IAM yang menambahkan konfigurasi analitik Anda harus memiliki [CreateServiceLinkedRole](#) izin. Peran terkait

layanan adalah. [AWSServiceRoleForAmazonCognitoIdp](#) Untuk informasi selengkapnya, lihat [Menggunakan peran terkait layanan untuk Amazon Cognito](#).

Saat menerapkan `AnalyticsConfiguration` ke klien aplikasi di Amazon Cognito API, Anda dapat menetapkan peran IAM khusus untuk Amazon Pinpoint dan ID eksternal untuk mengambil peran tersebut. Peran harus mempercayai prinsipal `cognito-idp` layanan, dan jika kebijakan kepercayaan peran memerlukan ID eksternal, itu harus sesuai dengan `AnalyticsConfiguration`. Anda harus memberikan `cognito-idp:Describe*` izin peran, dan izin berikut untuk proyek Amazon Pinpoint Anda.

- `mobiletargeting:UpdateEndpoint`
- `mobiletargeting:PutEvents`

## Ketersediaan Amazon Cognito dan Amazon Pinpoint Region

Tabel berikut menunjukkan Wilayah AWS pemetaan antara Amazon Cognito dan Amazon Pinpoint yang memenuhi salah satu kondisi berikut.

- Anda hanya dapat menggunakan proyek Amazon Pinpoint di Wilayah AS Timur (Virginia Utara) (`us-timur-1`).
- Anda dapat menggunakan proyek Amazon Pinpoint di Wilayah yang sama atau di Wilayah AS Timur (Virginia N.) (`us-timur-1`)

Secara default, Amazon Cognito hanya dapat mengirim analitik ke proyek Amazon Pinpoint dalam hal yang sama. Wilayah AWS Pengecualian untuk aturan ini adalah Wilayah dalam tabel berikut, dan Wilayah tempat Amazon Pinpoint tidak tersedia.

Amazon Pinpoint tidak tersedia di Wilayah berikut. Kumpulan pengguna Amazon Cognito di Wilayah ini tidak mendukung analitik.

- Europe (Milan)
- Middle East (Bahrain)
- Asia Pacific (Osaka)
- Israel (Tel Aviv)
- Africa (Cape Town)
- Asia Pacific (Jakarta)

Tabel menunjukkan hubungan antara Wilayah tempat Anda membangun kumpulan pengguna Amazon Cognito dan Wilayah terkait di Amazon Pinpoint. Anda harus mengonfigurasi proyek Amazon Pinpoint Anda di Wilayah yang tersedia untuk mengintegrasikannya dengan Amazon Cognito.

Wilayah kumpulan pengguna Amazon Cognito	Wilayah untuk proyek Amazon Pinpoint
ap-northeast-1	us-east-1
ap-northeast-2	us-east-1
ap-south-1	us-east-1, ap-south-1
ap-southeast-1	us-east-1
ap-southeast-2	us-east-1, ap-southeast-2
ca-central-1	us-east-1
eu-central-1	us-east-1, eu-central-1
eu-west-1	us-east-1, eu-west-1
eu-west-2	us-east-1
us-east-1	us-east-1
us-east-2	us-east-1
us-west-2	us-east-1, us-west-2

### Contoh pemetaan wilayah

- Jika Anda membuat kumpulan pengguna di ap-northeast-1, Anda dapat membuat proyek Amazon Pinpoint di us-east-1.
- Jika Anda membuat kumpulan pengguna di ap-selatan-1, Anda dapat membuat proyek Amazon Pinpoint di us-east-1 atau ap-south-1.

**Note**

Untuk semua Wilayah AWS kecuali yang ada di tabel sebelumnya, Amazon Cognito hanya dapat menggunakan proyek Amazon Pinpoint di Wilayah yang sama dengan kumpulan pengguna Anda. Jika Amazon Pinpoint tidak tersedia di Wilayah tempat Anda membuat kumpulan pengguna, dan tidak tercantum dalam tabel, Amazon Cognito tidak mendukung analitik Amazon Pinpoint di Wilayah tersebut. Untuk Wilayah AWS informasi lebih lanjut, lihat [Amazon Pinpoint endpoint](#) dan kuota.

## Menentukan setelan analitik Amazon Pinpoint () AWS Management Console

Anda dapat mengonfigurasi kumpulan pengguna Amazon Cognito untuk mengirim data analitik ke Amazon Pinpoint. Amazon Cognito hanya mengirimkan data analitik ke Amazon Pinpoint untuk pengguna lokal. Setelah mengonfigurasi kumpulan pengguna untuk diasosiasikan dengan project Amazon Pinpoint, Anda harus menyertakan AnalyticsMetadata permintaan API. Untuk informasi selengkapnya, lihat [Mengintegrasikan aplikasi Anda dengan Amazon Pinpoint](#).

Untuk menentukan pengaturan analitik

1. Masuk ke [Konsol Amazon Cognito](#). Anda mungkin diminta untuk kredensial AWS Anda.
2. Pilih Kumpulan Pengguna dan pilih kumpulan pengguna yang ada dari daftar.
3. Pilih tab Integrasi aplikasi.
4. Di bawah Klien dan analitik aplikasi, pilih nama klien App yang ada dari daftar.
5. Di bawah Pinpoint analytics, pilih Aktifkan.
6. Pilih Wilayah Pinpoint.
7. Pilih proyek Amazon Pinpoint atau pilih Buat proyek Amazon Pinpoint.

**Note**


ID proyek Amazon Pinpoint adalah string 32 karakter yang unik untuk proyek Amazon Pinpoint Anda. Itu dicantumkan di konsol Amazon Pinpoint.

Anda dapat memetakan beberapa aplikasi Amazon Cognito ke proyek Amazon Pinpoint tunggal. Namun, setiap aplikasi Amazon Cognito hanya dapat dipetakan ke satu proyek Amazon Pinpoint.

Di Amazon Pinpoint, setiap proyek harus menjadi satu aplikasi. Misalnya, jika developer game memiliki dua game, setiap game harus menjadi proyek Amazon Pinpoint yang

terpisah, bahkan jika kedua game menggunakan kolam pengguna Amazon Cognito yang sama. Untuk informasi lebih lanjut tentang proyek Amazon Pinpoint, lihat [Buat proyek di Amazon Pinpoint](#).

8. Di bawah Berbagi data pengguna, pilih Bagikan data pengguna dengan Amazon Pinpoint jika Anda ingin Amazon Cognito mengirim alamat email dan nomor telepon ke Amazon Pinpoint dan membuat titik akhir tambahan untuk pengguna. Setelah pengguna memverifikasi alamat email dan nomor telepon mereka, Amazon Cognito hanya membagikannya dengan Amazon Pinpoint jika tersedia untuk akun pengguna.

 Note

Sebuah titik akhir secara unik mengidentifikasi perangkat pengguna tempat Anda dapat mengirim notifikasi push dengan Amazon Pinpoint. Untuk informasi selengkapnya tentang titik akhir, lihat [Menambahkan titik akhir](#) di Panduan Pengembang Amazon Pinpoint.

9. Pilih Simpan perubahan.

## Menentukan setelan analitik Amazon Pinpoint AWS CLI (AWS dan API)

Gunakan perintah berikut untuk menentukan pengaturan analitik Amazon Pinpoint untuk kolam pengguna Anda.

Untuk menentukan pengaturan analitik untuk aplikasi klien kolam pengguna yang ada pada saat pembuatan aplikasi

- AWS CLI: `aws cognito-idp create-user-pool-client`
- AWSAPI: [CreateUserPoolClient](#)

Untuk memperbarui pengaturan analitik untuk aplikasi klien kolam pengguna yang sudah ada

- AWS CLI: `aws cognito-idp update-user-pool-client`
- AWSAPI: [UpdateUserPoolClient](#)

**Note**

Amazon Cognito mendukung integrasi di Wilayah saat Anda menggunakan `ApplicationArn`

## Mengintegrasikan aplikasi Anda dengan Amazon Pinpoint

Anda dapat mempublikasikan metadata analitik ke Amazon Pinpoint untuk pengguna lokal Amazon Cognito di API kumpulan pengguna.

### Pengguna lokal

Pengguna yang mendaftar akun atau dibuat di kumpulan pengguna Anda alih-alih masuk melalui penyedia identitas pihak ketiga (iDP).

### API kumpulan pengguna

Operasi yang dapat Anda integrasikan dengan AWS SDK, menggunakan aplikasi dengan antarmuka pengguna kustom (UI). Anda tidak dapat meneruskan metadata analitik untuk pengguna federasi atau lokal yang masuk melalui UI yang dihosting. Lihat [Referensi API Amazon Cognito](#) untuk mengetahui daftar operasi API kumpulan pengguna.

Setelah mengonfigurasi kumpulan pengguna untuk dipublikasikan ke kampanye, Amazon Cognito meneruskan metadata ke Amazon Pinpoint untuk operasi API berikut.

- `AdminInitiateAuth`
- `AdminRespondToAuthChallenge`
- `ConfirmForgotPassword`
- `ConfirmSignUp`
- `ForgotPassword`
- `InitiateAuth`
- `ResendConfirmationCode`
- `RespondToAuthChallenge`
- `SignUp`



Untuk meneruskan metadata tentang sesi pengguna ke kampanye Amazon Pinpoint, sertakan `AnalyticsEndpointId` nilai dalam parameter permintaan `AnalyticsMetadata` API Anda. Sebagai JavaScript contoh, lihat [Mengapa analitik kumpulan pengguna Amazon Cognito saya tidak muncul di dasbor Amazon Pinpoint saya?](#) di pusat AWS pengetahuan.

## Mengkonfigurasi analisis kumpulan pengguna

Menggunakan Analitik Amazon Pinpoint, Anda dapat melacak pendaftaran, proses masuk, autentikasi yang gagal, pengguna aktif harian (DAU), dan pengguna aktif bulanan (MAU) kolom pengguna Amazon Cognito. Anda juga dapat menyiapkan atribut pengguna tertentu untuk aplikasi Anda menggunakan AWS Mobile SDK for Android atau AWS Mobile SDK for iOS. Itu kemudian dapat digunakan untuk menyegmentasikan pengguna Anda di Amazon Pinpoint dan mengirim mereka notifikasi push yang ditargetkan.

Di tab Integrasi aplikasi di bawah Klien aplikasi dan analitik, Anda dapat menavigasi ke klien aplikasi yang sudah ada atau membuat klien baru. Dalam konfigurasi klien aplikasi, Anda dapat menentukan project Amazon Pinpoint yang ingin digunakan dengan aplikasi Anda. Untuk informasi selengkapnya, lihat [Menggunakan Analitik Amazon Pinpoint dengan kolam pengguna Amazon Cognito](#).

### Note

Amazon Pinpoint tersedia dalam beberapa Daerah AWS di Amerika Utara, Eropa, Asia, dan Oseania. Amazon Pinpoint Regions menyertakan Amazon Pinpoint API. Jika wilayah Amazon Pinpoint didukung oleh Amazon Cognito, Amazon Cognito akan mengirim acara ke proyek Amazon Pinpoint dalam wilayah Amazon Pinpoint yang sama. Jika sebuah wilayah tidak didukung oleh Amazon Pinpoint, maka Amazon Cognito hanya akan mendukung mengirim peristiwa di `us-east-1`. Untuk informasi wilayah terperinci Amazon Pinpoint, lihat titik akhir dan kuota [Amazon Pinpoint dan Menggunakan analitik Amazon Pinpoint dengan kumpulan pengguna Amazon Cognito](#).

Untuk menambahkan analitik dan kampanye

1. Pilih Tambahkan analitik dan kampanye.
2. Pilih sebuah Klien aplikasi Cognito dari daftar.
3. Untuk memetakan aplikasi Amazon Cognito Anda ke sebuah proyek Amazon Pinpoint, pilih proyek Amazon Pinpoint dari daftar.

**Note**

ID proyek Amazon Pinpoint adalah string 32 karakter yang unik untuk proyek Amazon Pinpoint Anda. Itu mencantumkan konsol Amazon Pinpoint.

Anda dapat memetakan beberapa aplikasi Amazon Cognito ke proyek Amazon Pinpoint tunggal. Namun, setiap aplikasi Amazon Cognito hanya dapat dipetakan ke satu proyek Amazon Pinpoint.

Di Amazon Pinpoint, setiap proyek harus menjadi satu aplikasi. Misalnya, jika developer game memiliki dua game, setiap game harus menjadi proyek Amazon Pinpoint yang terpisah, bahkan jika kedua game menggunakan kolam pengguna Amazon Cognito yang sama.

4. Pilih Bagikan data atribut pengguna dengan Amazon Pinpoint jika Anda ingin Amazon Cognito mengirim alamat email dan nomor telepon ke Amazon Pinpoint untuk membuat titik akhir tambahan bagi pengguna.

**Note**

Endpoint secara unik mengidentifikasi perangkat pengguna tempat Anda dapat mengirim pemberitahuan push dengan Amazon Pinpoint. Untuk informasi selengkapnya tentang titik akhir, lihat [Menambahkan titik akhir ke Amazon Pinpoint di](#) Panduan Pengembang Amazon Pinpoint.

5. Masukkan IAM role yang telah Anda buat atau pilih Buat peran baru untuk membuat peran baru di konsol IAM.
6. Pilih Simpan perubahan.
7. Untuk menentukan pemetaan aplikasi tambahan, pilih Tambahkan pemetaan aplikasi.
8. Pilih Simpan perubahan.

## Mengelola pengguna di kumpulan pengguna Anda

Setelah membuat kumpulan pengguna, Anda dapat membuat, mengonfirmasi, dan mengelola akun pengguna. Dengan grup kolam pengguna Amazon Cognito Anda dapat mengelola pengguna Anda dan akses mereka ke sumber daya dengan memetakan peran IAM ke grup.

Anda dapat mengimpor pengguna Anda ke sebuah kolam pengguna dengan pemicu Lambda migrasi pengguna. Pendekatan ini memungkinkan migrasi pengguna yang mulus dari direktori pengguna Anda yang sudah ada ke kolam pengguna ketika mereka masuk ke kolam pengguna Anda pertama kali.

## Topik

- [Mengkonfigurasi kebijakan untuk pembuatan pengguna](#)
- [Mendaftar dan mengonfirmasi akun pengguna](#)
- [Membuat akun pengguna sebagai administrator](#)
- [Menambahkan grup ke kumpulan pengguna](#)
- [Mengelola dan mencari akun pengguna](#)
- [Memulihkan akun pengguna](#)
- [Mengimpor pengguna ke kolam pengguna](#)
- [Atribut kumpulan pengguna](#)
- [Menambahkan persyaratan kata sandi kumpulan pengguna](#)

## Mengkonfigurasi kebijakan untuk pembuatan pengguna

Kumpulan pengguna Anda dapat mengizinkan pengguna untuk mendaftar, atau Anda dapat membuatnya sebagai administrator. Anda juga dapat mengontrol seberapa banyak proses verifikasi dan konfirmasi setelah pendaftaran berada di tangan pengguna Anda. Misalnya, Anda mungkin ingin meninjau pendaftaran dan menerimanya berdasarkan proses validasi eksternal. Konfigurasi ini, atau admin membuat kebijakan pengguna, juga menetapkan jumlah waktu sebelum pengguna tidak dapat lagi mengonfirmasi akun penggunanya.

Amazon Cognito dapat melayani kebutuhan pelanggan publik Anda sebagai platform identitas pelanggan dan manajemen akses (CIAM) untuk perangkat lunak Anda. Kumpulan pengguna yang menerima pendaftaran dan memiliki klien aplikasi, dengan atau tanpa UI yang dihosting, membuat profil pengguna untuk siapa saja di internet yang mengetahui ID klien aplikasi Anda yang dapat ditemukan secara publik dan permintaan untuk mendaftar. Profil pengguna yang terdaftar dapat menerima akses dan token identitas dan dapat mengakses sumber daya yang telah Anda otorisasi untuk aplikasi Anda. Sebelum Anda mengaktifkan pendaftaran di kumpulan pengguna, tinjau opsi Anda dan pastikan konfigurasi Anda sesuai dengan standar keamanan Anda. Set Aktifkan pendaftaran mandiri dan `AllowAdminCreateUserOnly`, dijelaskan dalam prosedur berikut, dengan hati-hati.

## AWS Management Console

Tab Pengalaman pendaftaran kumpulan pengguna Anda dan langkah Konfigurasi pengalaman pendaftaran dari wizard buat kumpulan pengguna berisi beberapa pengaturan untuk pendaftaran dan pembuatan administratif pengguna di kumpulan pengguna Anda.

Untuk mengonfigurasi pengalaman pendaftaran

1. Dalam verifikasi dan konfirmasi yang dibantu Cognito, pilih apakah Anda ingin Izinkan Cognito mengirim pesan secara otomatis untuk memverifikasi dan mengonfirmasi. Dengan pengaturan ini diaktifkan, Amazon Cognito mengirimkan email atau pesan SMS ke pengguna baru dengan kode yang harus mereka tunjukkan ke kumpulan pengguna Anda. Ini mengonfirmasi kepemilikan mereka atas alamat email atau nomor telepon, menyetel atribut yang setara sebagai terverifikasi dan mengonfirmasi akun pengguna untuk masuk. Atribut untuk memverifikasi bahwa Anda memilih menentukan metode pengiriman dan tujuan pesan verifikasi.
2. Memverifikasi perubahan atribut tidak signifikan saat Anda membuat pengguna, tetapi terkait dengan verifikasi atribut. Anda dapat mengizinkan pengguna yang telah mengubah tetapi belum memverifikasi [atribut login](#) mereka untuk terus masuk baik dengan nilai atribut baru atau dengan aslinya. Untuk informasi selengkapnya, lihat [Memverifikasi kapan pengguna mengubah email atau nomor telepon mereka](#).
3. Atribut wajib menampilkan atribut yang harus diberikan nilai sebelum pengguna dapat mendaftar atau Anda dapat membuat pengguna. Anda hanya dapat mengatur atribut yang diperlukan di wizard buat kumpulan pengguna.
4. Atribut khusus penting untuk proses pembuatan dan pendaftaran pengguna karena Anda hanya dapat menetapkan nilai untuk atribut kustom yang tidak dapat diubah saat pertama kali membuat pengguna. Untuk informasi selengkapnya tentang atribut kustom, lihat [Atribut kustom](#).
5. [Dalam Pendaftaran layanan mandiri, pilih Aktifkan pendaftaran mandiri jika Anda ingin pengguna dapat membuat akun baru dengan API yang tidak diautentikasi](#). `SignUp` Jika menonaktifkan pendaftaran mandiri, Anda hanya dapat membuat pengguna baru sebagai administrator, di konsol Amazon Cognito, atau `AdminCreateUser` dengan permintaan API. Di kumpulan pengguna di mana pendaftaran mandiri tidak aktif, permintaan `SignUp` API kembali `NotAuthorizedException` dan UI yang dihosting tidak menampilkan tautan Daftar.

Untuk kumpulan pengguna tempat Anda berencana membuat pengguna sebagai administrator, Anda dapat mengonfigurasi durasi kata sandi sementara mereka di tab Pengalaman masuk di bawah Kata sandi sementara yang ditetapkan oleh administrator akan kedaluwarsa.

Elemen penting lainnya dari penciptaan pengguna sebagai administrator adalah pesan undangan. Saat Anda membuat pengguna baru, Amazon Cognito mengiriminya mereka pesan dengan tautan ke aplikasi Anda sehingga mereka dapat masuk untuk pertama kalinya. Sesuaikan templat pesan ini di tab Pesan di bawah Templat Pesan.

Anda dapat mengonfigurasi [klien aplikasi rahasia](#), biasanya aplikasi web, dengan rahasia klien yang mencegah pendaftaran tanpa rahasia klien aplikasi. Sebagai praktik keamanan terbaik, jangan mendistribusikan rahasia klien aplikasi di klien aplikasi publik, biasanya aplikasi seluler. Anda dapat membuat klien aplikasi dengan rahasia klien di tab Integrasi aplikasi di konsol Amazon Cognito.

## Amazon Cognito user pools API

Anda dapat mengatur parameter untuk pembuatan pengguna secara terprogram di kumpulan pengguna dalam permintaan [CreateUserPool](#) atau [UpdateUserPool](#) API.

[AdminCreateUserConfig](#) Elemen menetapkan nilai untuk properti berikut dari kumpulan pengguna.

1. Aktifkan pendaftaran swalayan
2. Pesan undangan yang Anda kirim ke pengguna baru yang dibuat admin

Contoh berikut, ketika ditambahkan ke badan permintaan API lengkap, menetapkan kumpulan pengguna dengan pendaftaran swalayan tidak aktif dan email undangan dasar.

```
"AdminCreateUserConfig": {
  "AllowAdminCreateUserOnly": true,
  "InviteMessageTemplate": {
    "EmailMessage": "Your username is {username} and temporary password is
    {#####}.",
    "EmailSubject": "Welcome to ExampleApp",
    "SMSMessage": "Your username is {username} and temporary password is
    {#####}."
  }
}
```

Parameter tambahan berikut dari permintaan [CreateUserPool](#) atau [UpdateUserPool](#) API mengatur pembuatan pengguna baru.

## [AutoVerifiedAttributes](#)

Atribut, alamat email, atau nomor telepon, yang ingin Anda [kirimi pesan secara otomatis](#) saat Anda mendaftarkan pengguna baru.

## [Kebijakan](#)

[Kebijakan kata sandi](#) kumpulan pengguna.

## [Skema](#)

[Atribut kustom](#) kumpulan pengguna. Mereka penting untuk proses pembuatan dan pendaftaran pengguna karena Anda hanya dapat menetapkan nilai untuk atribut kustom yang tidak dapat diubah saat pertama kali membuat pengguna.

Parameter ini juga menetapkan atribut yang diperlukan untuk kumpulan pengguna Anda. Teks berikut, ketika dimasukkan ke dalam Schema elemen badan permintaan API lengkap, mengatur email atribut seperti yang diperlukan.

```
{
    "Name": "email",
    "Required": true
}
```

## Mendaftar dan mengonfirmasi akun pengguna

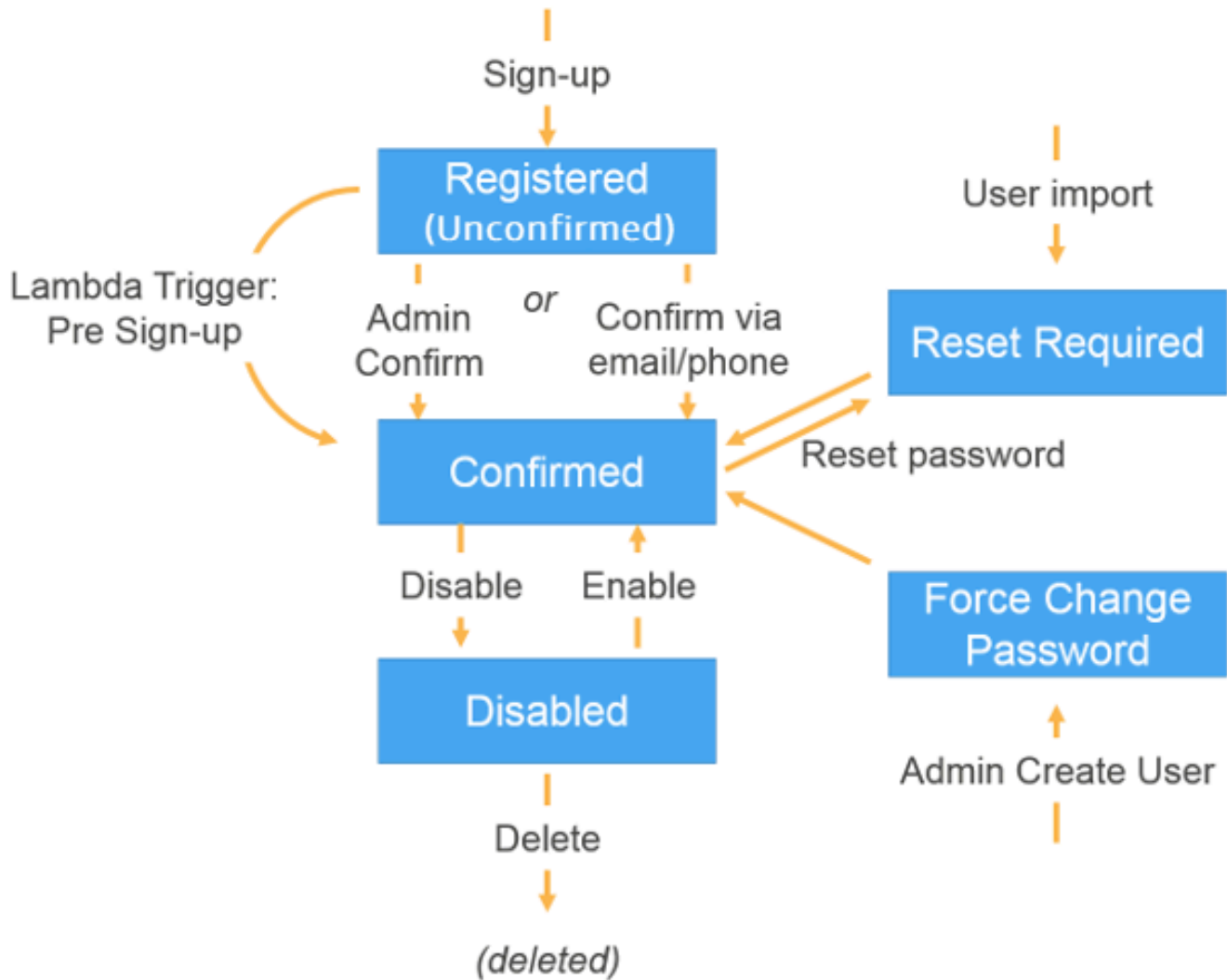
Akun pengguna ditambahkan ke kolam pengguna Anda dengan salah satu cara berikut:

- Pengguna mendaftar di aplikasi klien kumpulan pengguna Anda. Ini bisa berupa aplikasi seluler atau web.
- Anda dapat mengimpor akun pengguna ke dalam kolam pengguna Anda. Untuk informasi lebih lanjut, lihat [Mengimpor pengguna ke kumpulan pengguna dari file CSV](#).
- Anda dapat membuat akun pengguna di kolam pengguna Anda dan mengundang pengguna untuk masuk. Untuk informasi selengkapnya, lihat [Membuat akun pengguna sebagai administrator](#).

Pengguna yang mendaftar sendiri harus dikonfirmasi sebelum mereka dapat masuk. Pengguna yang diimpor dan dibuat sudah dikonfirmasi, tetapi mereka harus membuat kata sandi saat pertama kali masuk. Bagian berikut menjelaskan proses konfirmasi dan verifikasi email dan telepon.

## Ikhtisar konfirmasi akun pengguna

Diagram berikut menggambarkan proses konfirmasi:



Akun pengguna dapat berada di salah satu status berikut:

### Terdaftar (Belum Dikonfirmasi)

Pengguna telah berhasil mendaftar, tetapi tidak dapat masuk sebelum akun pengguna dikonfirmasi. Pengguna telah diaktifkan tetapi tidak dikonfirmasi dalam status ini.

Pengguna baru yang mendaftar mulai dari status ini.

## Terkonfirmasi

Akun pengguna dikonfirmasi dan pengguna dapat masuk. Ketika pengguna memasukkan kode atau mengikuti tautan email untuk mengonfirmasi akun pengguna mereka, email atau nomor telepon tersebut diverifikasi secara otomatis. Kode atau tautan verifikasi valid selama 24 jam.

Jika akun pengguna dikonfirmasi oleh administrator atau pemicu Lambda sebelum pendaftaran, mungkin tidak ada email atau nomor telepon terverifikasi yang terkait dengan akun tersebut.

## Atur Ulang Kata Sandi Diperlukan

Akun pengguna dikonfirmasi, tetapi pengguna harus meminta kode dan mengatur ulang kata sandi mereka sebelum mereka dapat masuk.

Akun pengguna yang diimpor oleh administrator atau developer mulai di keadaan ini.

## Paksa Ubah Kata Sandi

Akun pengguna dikonfirmasi dan pengguna dapat masuk menggunakan kata sandi sementara, tetapi saat masuk pertama, pengguna harus mengubah kata sandi mereka ke nilai baru sebelum melakukan hal lain.

Akun pengguna yang diimpor oleh administrator atau developer yang di mulai pada status ini.

## Nonaktif

Sebelum Anda dapat menghapus akun pengguna, Anda harus menonaktifkan akses masuk untuk pengguna tersebut.

## Memverifikasi informasi kontak saat mendaftar

Ketika pengguna baru mendaftar di aplikasi Anda, Anda mungkin ingin mereka untuk menyediakan paling tidak satu metode kontak. Misalnya, dengan informasi kontak pengguna Anda, Anda mungkin:

- Kirim kata sandi sementara ketika pengguna memilih untuk mengatur ulang kata sandi mereka.
- Beri tahu pengguna ketika informasi personal atau keuangan mereka diperbarui.
- Kirim pesan promosi, seperti penawaran khusus atau diskon.
- Kirim ringkasan akun atau pengingat penagihan.

Untuk kasus penggunaan seperti ini, Anda harus mengirim pesan ke tujuan yang terverifikasi. Jika tidak, Anda mungkin mengirim pesan Anda ke alamat email yang tidak valid atau nomor telepon yang



tidak diketik dengan benar. Atau lebih buruk lagi, Anda mungkin mengirim informasi sensitif ke aktor jahat yang menyamar sebagai pengguna Anda.

Untuk membantu memastikan bahwa Anda hanya mengirim pesan ke individu yang tepat, konfigurasi kolam pengguna Amazon Cognito Anda sehingga pengguna harus menyediakan hal berikut saat mereka mendaftar:

- a. Alamat email atau nomor telepon.
- b. Kode verifikasi yang dikirim Amazon Cognito ke alamat email atau nomor telepon tersebut.  
Jika 24 jam telah berlalu dan kode atau tautan pengguna Anda tidak lagi valid, hubungi operasi [ResendConfirmationCode](#) API untuk membuat dan mengirim kode atau tautan baru.

Dengan memberikan kode verifikasi, pengguna membuktikan bahwa mereka memiliki akses ke kotak surat atau telepon yang menerima kode tersebut. Setelah pengguna memberikan kode, Amazon Cognito memperbarui informasi tentang pengguna di kolam pengguna Anda dengan:

- Mengatur status pengguna ke CONFIRMED.
- Memperbarui atribut pengguna untuk menunjukkan bahwa alamat email atau nomor telepon terverifikasi.

Untuk melihat informasi ini, Anda dapat menggunakan konsol Amazon Cognito. Atau, Anda dapat menggunakan operasi `AdminGetUser` API, `admin-get-user` perintah dengan AWS CLI, atau tindakan yang sesuai di salah satu AWS SDK.

Jika pengguna memiliki metode kontak terverifikasi, Amazon Cognito secara otomatis mengirim pesan ke pengguna saat pengguna meminta pengaturan ulang kata sandi.

Untuk mengonfigurasi kumpulan pengguna Anda agar memerlukan verifikasi email atau telepon

Ketika Anda memverifikasi alamat email dan nomor telepon pengguna Anda, Anda memastikan bahwa Anda dapat menghubungi pengguna Anda. Selesaikan langkah-langkah berikut di AWS Management Console untuk mengonfigurasi kumpulan pengguna Anda agar pengguna Anda mengonfirmasi alamat email atau nomor telepon mereka.

#### Note

Jika Anda belum memiliki kumpulan pengguna di akun Anda, lihat [Memulai dengan kumpulan pengguna](#).

## Untuk mengonfigurasi kolam pengguna

1. Arahkan ke konsol [Amazon Cognito](#). Jika diminta, masukkan AWS kredensial Anda.
2. Dari panel navigasi, pilih User Pools. Pilih kolam pengguna yang ada dari daftar, atau [buat kolam pengguna](#).
3. Pilih tab Sign-up experience dan temukan Verifikasi atribut dan konfirmasi akun pengguna. Pilih Edit.
4. Di bawah verifikasi dan konfirmasi yang dibantu Cognito, pilih apakah Anda akan Izinkan Cognito mengirim pesan secara otomatis untuk memverifikasi dan mengonfirmasi. Dengan pengaturan ini diaktifkan, Amazon Cognito mengirimkan pesan ke atribut kontak pengguna yang Anda pilih saat pengguna mendaftar, atau Anda membuat profil pengguna. Untuk memverifikasi atribut dan mengonfirmasi profil pengguna untuk masuk, Amazon Cognito mengirimkan kode atau tautan dalam pesan ke pengguna. Pengguna kemudian harus memasukkan kode di UI Anda sehingga aplikasi Anda dapat mengonfirmasinya dalam permintaan `ConfirmSignUp` atau `AdminConfirmSignUp` API.

### Note

Anda juga dapat menonaktifkan verifikasi dan konfirmasi yang dibantu Cognito dan menggunakan tindakan API yang diautentikasi atau pemicu Lambda untuk memverifikasi atribut dan mengonfirmasi pengguna.

Jika Anda memilih opsi ini, Amazon Cognito tidak mengirim email kode verifikasi ketika pengguna mendaftar. Pilih opsi ini jika Anda menggunakan alur autentikasi kustom yang memverifikasi setidaknya satu metode kontak tanpa menggunakan kode verifikasi dari Amazon Cognito. Sebagai contoh, Anda mungkin menggunakan pemicu pre sign-up Lambda yang secara otomatis memverifikasi alamat email yang termasuk ke domain spesifik.

Jika Anda tidak memverifikasi informasi kontak pengguna, mereka mungkin tidak dapat menggunakan aplikasi Anda. Ingat bahwa pengguna memerlukan informasi kontak terverifikasi untuk:

- Setel ulang kata sandi mereka — Saat pengguna memilih opsi di aplikasi Anda yang memanggil tindakan `ForgotPassword` API, Amazon Cognito mengirimkan kata sandi sementara ke alamat email atau nomor telepon pengguna. Amazon Cognito mengirim kata sandi ini hanya jika pengguna memiliki setidaknya satu metode kontak terverifikasi.

- Masuk dengan menggunakan alamat email atau nomor telepon sebagai alias — Jika Anda mengonfigurasi kumpulan pengguna untuk mengizinkan alias ini, maka pengguna dapat masuk dengan alias hanya jika alias diverifikasi. Untuk informasi selengkapnya, lihat [Menyesuaikan atribut masuk](#).

## 5. Pilih Atribut Anda untuk memverifikasi:

### Kirim pesan SMS, verifikasi nomor telepon

Amazon Cognito mengirimkan kode verifikasi dalam pesan SMS saat pengguna mendaftar. Pilih opsi ini jika Anda biasanya berkomunikasi dengan pengguna melalui pesan SMS. Sebagai contoh, Anda akan ingin menggunakan nomor telepon terverifikasi jika Anda mengirim pemberitahuan pengiriman, konfirmasi janji temu, atau peringatan. Nomor telepon pengguna akan menjadi atribut terverifikasi ketika akun dikonfirmasi; Anda harus mengambil tindakan tambahan untuk memverifikasi dan berkomunikasi dengan alamat email pengguna.

### Kirim pesan email, verifikasi alamat email

Amazon Cognito mengirimkan kode verifikasi melalui pesan email saat pengguna mendaftar. Pilih opsi ini jika Anda biasanya berkomunikasi dengan pengguna Anda melalui email. Sebagai contoh, Anda akan ingin menggunakan alamat email terverifikasi jika Anda mengirimkan laporan tagihan, ringkasan pesanan, atau penawaran khusus. Alamat email pengguna akan menjadi atribut terverifikasi ketika akun dikonfirmasi; Anda harus mengambil tindakan tambahan untuk memverifikasi dan berkomunikasi dengan nomor telepon pengguna.

### Kirim pesan SMS jika nomor telepon tersedia, jika tidak, kirim pesan email

Pilih opsi ini jika Anda tidak memerlukan semua pengguna untuk memiliki metode kontak terverifikasi yang sama. Dalam kasus ini, halaman pendaftaran di aplikasi Anda dapat meminta pengguna untuk memverifikasi hanya metode kontak yang mereka sukai. Ketika Amazon Cognito mengirimkan kode verifikasi, ia mengirimkan kode ke metode kontak yang disediakan di permintaan `SignUp` dari aplikasi Anda. Jika pengguna memberikan alamat email dan nomor telepon, dan aplikasi Anda menyediakan kedua metode kontak di permintaan `SignUp`, Amazon Cognito mengirimkan kode verifikasi hanya ke nomor telepon.

Jika Anda mengharuskan pengguna untuk memverifikasi alamat email dan nomor telepon, pilih opsi ini. Amazon Cognito memverifikasi satu metode kontak saat pengguna mendaftar, dan aplikasi Anda harus memverifikasi metode kontak lainnya setelah pengguna masuk.

Untuk informasi selengkapnya, lihat [Jika Anda meminta pengguna untuk mengonfirmasi alamat email dan nomor telepon](#).

## 6. Pilih Simpan perubahan.

Alur otentikasi dengan verifikasi email atau telepon

Jika kolam pengguna Anda memerlukan pengguna untuk memverifikasi informasi kontak mereka, aplikasi Anda harus memfasilitasi alur berikut ini ketika pengguna mendaftar:

1. Pengguna mendaftar di aplikasi Anda dengan memasukkan nama pengguna, nomor telepon dan/atau alamat email, dan mungkin atribut lain.
2. Layanan Amazon Cognito menerima permintaan pendaftaran dari aplikasi. Setelah memverifikasi bahwa permintaan berisi semua atribut yang diperlukan untuk pendaftaran, layanan menyelesaikan proses pendaftaran dan mengirimkan kode konfirmasi ke telepon pengguna (dalam pesan SMS) atau email. Kode ini valid untuk 24 jam.
3. Layanan mengembalikan ke aplikasi bahwa pendaftaran telah selesai dan akun pengguna sedang menunggu konfirmasi. Tanggapan berisi informasi tentang di mana kode konfirmasi dikirim. Pada titik ini akun pengguna berada dalam keadaan tidak terkonfirmasi, dan alamat email dan nomor telepon pengguna tidak terverifikasi.
4. Aplikasi sekarang dapat meminta pengguna untuk memasukkan kode konfirmasi. Pengguna tidak diharuskan untuk segera memasukkan kode. Namun, pengguna tidak akan bisa masuk sampai setelah mereka memasukkan kode konfirmasi.
5. Pengguna memasukkan kode konfirmasi di dalam aplikasi.
6. Aplikasi memanggil [ConfirmSignUp](#) untuk mengirim kode ke layanan Amazon Cognito, yang memverifikasi kode dan, jika kode benar, menetapkan akun pengguna ke status terkonfirmasi. Setelah berhasil mengonfirmasi akun pengguna, layanan Amazon Cognito secara otomatis menandai atribut yang digunakan untuk mengonfirmasi (alamat email atau nomor telepon) sebagai diverifikasi. Kecuali nilai atribut ini berubah, pengguna tidak perlu memverifikasinya lagi.
7. Pada titik ini akun pengguna dalam keadaan dikonfirmasi, dan pengguna dapat masuk.

Jika Anda meminta pengguna untuk mengonfirmasi alamat email dan nomor telepon

Amazon Cognito memverifikasi hanya satu metode kontak ketika pengguna mendaftar. Dalam kasus di mana Amazon Cognito harus memilih antara memverifikasi alamat email atau nomor telepon, ia memilih untuk memverifikasi nomor telepon dengan mengirimkan kode verifikasi melalui pesan SMS. Sebagai contoh, jika Anda mengonfigurasi kolam pengguna Anda untuk mengizinkan

pengguna untuk memverifikasi alamat email atau nomor telepon, dan jika aplikasi Anda menyediakan semua dari atribut ini saat pendaftaran, Amazon Cognito hanya memverifikasi nomor telepon. Setelah pengguna memverifikasi nomor teleponnya, Amazon Cognito menyetel status CONFIRMED pengguna, dan pengguna diizinkan untuk masuk ke aplikasi Anda.

Setelah pengguna masuk, aplikasi Anda dapat menyediakan opsi untuk memverifikasi metode kontak yang belum terverifikasi selama pendaftaran. Untuk memverifikasi metode kedua ini, aplikasi Anda memanggil tindakan API `VerifyUserAttribute`. Perhatikan bahwa tindakan ini memerlukan parameter `AccessToken`, dan Amazon Cognito hanya menyediakan token akses untuk pengguna terautentikasi. Karena itu, Anda dapat memverifikasi metode kontak kedua hanya setelah pengguna masuk.

Jika Anda mengharuskan pengguna untuk memverifikasi alamat email dan nomor telepon, lakukan berikut ini:

1. Konfigurasi kolom pengguna Anda untuk mengizinkan pengguna untuk memverifikasi alamat email atau nomor telepon.
2. Dalam alur pendaftaran untuk aplikasi Anda, haruskan pengguna untuk memberikan alamat email dan nomor telepon. Panggil tindakan API [SignUp](#), dan sediakan alamat email atau nomor telepon untuk parameter `UserAttributes`. Pada titik ini, Amazon Cognito mengirim kode verifikasi ke telepon pengguna.
3. Di antarmuka aplikasi Anda, tunjukkan halaman konfirmasi di mana pengguna memasukkan kode verifikasi. Konfirmasi pengguna dengan memanggil tindakan API [ConfirmSignUp](#). Pada titik ini, status pengguna adalah CONFIRMED, dan nomor telepon pengguna terverifikasi, tetapi alamat email tidak terverifikasi.
4. Tampilkan halaman masuk, dan autentikasi pengguna dengan memanggil tindakan API [InitiateAuth](#). Setelah pengguna terkonfirmasi, Amazon Cognito mengembalikan token akses ke aplikasi Anda.
5. Panggil tindakan API [GetUserAttributeVerificationCode](#). Tentukan parameter berikut di permintaan:
  - `AccessToken` – Token akses yang dikembalikan oleh Amazon Cognito saat pengguna masuk.
  - `AttributeName` – Tentukan "email" sebagai nilai atribut.

Amazon Cognito mengirimkan kode verifikasi ke alamat email pengguna.

6. Tunjukkan halaman konfirmasi di mana pengguna memasukkan kode verifikasi. Ketika pengguna memasukkan kode, panggil tindakan API [VerifyUserAttribute](#). Tentukan parameter berikut di permintaan:
  - `AccessToken` – Token akses yang dikembalikan oleh Amazon Cognito saat pengguna masuk.
  - `AttributeName` – Tentukan "email" sebagai nilai atribut.
  - `Code` – Kode verifikasi yang diberikan pengguna.

Pada titik ini, alamat email terverifikasi.

## Memungkinkan pengguna untuk mendaftar di aplikasi Anda tetapi mengonfirmasinya sebagai administrator kumpulan pengguna

Anda mungkin tidak ingin kumpulan pengguna Anda secara otomatis mengirim pesan verifikasi di kumpulan pengguna Anda, tetapi tetap ingin mengizinkan siapa pun untuk mendaftar akun. Model ini menyisakan ruang, misalnya, untuk tinjauan manusia atas permintaan pendaftaran baru, dan untuk validasi batch dan pemrosesan pendaftaran. Anda dapat mengonfirmasi akun pengguna baru di konsol Amazon Cognito atau dengan operasi API yang diautentikasi oleh IAM. [AdminConfirmSignUp](#) Anda dapat mengonfirmasi akun pengguna sebagai administrator apakah kumpulan pengguna Anda mengirim pesan verifikasi atau tidak.

Anda hanya dapat mengonfirmasi pendaftaran layanan mandiri pengguna dengan teknik ini. Untuk mengonfirmasi pengguna yang Anda buat sebagai administrator, buat permintaan [AdminSetUserPassword](#) API dengan `Permanent` disetel ke `True`.

1. Pengguna mendaftar di aplikasi Anda dengan memasukkan nama pengguna, nomor telepon dan/atau alamat email, dan mungkin atribut lain.
2. Layanan Amazon Cognito menerima permintaan pendaftaran dari aplikasi. Setelah memverifikasi bahwa permintaan berisi semua atribut yang dibutuhkan untuk pendaftaran, layanan menyelesaikan proses pendaftaran mengembalikan ke aplikasi bahwa pendaftaran telah selesai, menunggu konfirmasi. Pada titik ini akun pengguna dalam keadaan belum dikonfirmasi. Pengguna tidak dapat masuk sebelum akun dikonfirmasi.
3. Konfirmasikan akun pengguna. Anda harus masuk ke AWS Management Console atau menandatangani permintaan API Anda dengan AWS kredensial untuk mengonfirmasi akun.

- a. Untuk mengonfirmasi pengguna di konsol Amazon Cognito, navigasikan ke tab Pengguna, pilih pengguna yang ingin Anda konfirmasi, dan dari menu Tindakan pilih Konfirmasi.
  - b. Untuk mengonfirmasi pengguna di AWS API atau CLI, buat permintaan [AdminConfirmSignUp](#) API, atau [admin-confirm-sign-up](#) di file. AWS CLI
4. Pada titik ini akun pengguna dalam keadaan dikonfirmasi, dan pengguna dapat masuk.

## Menghitung nilai hash rahasia

Tetapkan rahasia klien ke klien aplikasi rahasia Anda sebagai praktik terbaik. Saat Anda menetapkan rahasia klien ke klien aplikasi, permintaan API kumpulan pengguna Amazon Cognito Anda harus menyertakan hash yang menyertakan rahasia klien di badan permintaan. Untuk memvalidasi pengetahuan Anda tentang rahasia klien untuk operasi API dalam daftar berikut, menggabungkan rahasia klien dengan ID klien aplikasi dan nama pengguna pengguna Anda, lalu base64-encode string tersebut.

Saat aplikasi Anda menandatangani pengguna ke klien yang memiliki hash rahasia, Anda dapat menggunakan nilai atribut login kumpulan pengguna apa pun sebagai elemen nama pengguna dari hash rahasia. Saat aplikasi Anda meminta token baru dalam operasi autentikasi dengan `REFRESH_TOKEN_AUTH`, nilai elemen nama pengguna bergantung pada atribut login Anda. Jika kumpulan pengguna Anda tidak memiliki `username` atribut login, tetapkan nilai nama pengguna hash rahasia dari sub klaim pengguna dari akses atau token ID mereka. `username` kapan atribut login, atur nilai `username` hash rahasia dari klaim. `username`

API kumpulan pengguna Amazon Cognito berikut menerima nilai hash rahasia klien dalam suatu parameter. `SecretHash`

- [ConfirmForgotPassword](#)
- [ConfirmSignUp](#)
- [ForgotPassword](#)
- [ResendConfirmationCode](#)
- [SignUp](#)

Selain itu, API berikut menerima nilai hash rahasia klien dalam `SECRET_HASH` parameter, baik dalam parameter otentikasi atau dalam respons tantangan.

Operasi API	Parameter induk untuk SECRET_HASH
InitiateAuth	AuthParameters
AdminInitiateAuth	AuthParameters
RespondToAuthChallenge	ChallengeResponses
AdminRespondToAuthChallenge	ChallengeResponses

Nilai hash rahasia adalah kode otentikasi pesan hash kunci (HMAC) yang dikodekan Base 64 yang dihitung menggunakan kunci rahasia klien kumpulan pengguna dan nama pengguna ditambah ID klien dalam pesan. Kode pseudo berikut menunjukkan bagaimana nilai ini dihitung. Dalam kode pseudo ini, + menunjukkan kontegasi, HMAC\_SHA256 mewakili fungsi yang menghasilkan nilai HMAC menggunakan HMACSHA256, dan Base64 mewakili fungsi yang menghasilkan versi yang dienkodkan Base-64 dari output hash.

```
Base64 ( HMAC_SHA256 ( "Client Secret Key", "Username" + "Client Id" ) )
```

Untuk ikhtisar terperinci tentang cara menghitung dan menggunakan SecretHash parameter, lihat [Bagaimana cara memecahkan masalah kesalahan “Tidak dapat memverifikasi hash rahasia untuk klien” dari API kumpulan pengguna Amazon Cognito saya?](#) <client-id> di pusat AWS pengetahuan.

Anda dapat menggunakan contoh kode berikut dalam kode aplikasi sisi server Anda.

## Shell

```
echo -n "[username][app client ID]" | openssl dgst -sha256 -hmac [app client secret]
-binary | openssl enc -base64
```

## Java

```
import javax.crypto.Mac;
import javax.crypto.spec.SecretKeySpec;

public static String calculateSecretHash(String userPoolClientId, String
userPoolClientSecret, String userName) {
    final String HMAC_SHA256_ALGORITHM = "HmacSHA256";
```



```

    SecretKeySpec signingKey = new SecretKeySpec(
        userPoolClientSecret.getBytes(StandardCharsets.UTF_8),
        HMAC_SHA256_ALGORITHM);
    try {
        Mac mac = Mac.getInstance(HMAC_SHA256_ALGORITHM);
        mac.init(signingKey);
        mac.update(userName.getBytes(StandardCharsets.UTF_8));
        byte[] rawHmac =
    mac.doFinal(userPoolClientId.getBytes(StandardCharsets.UTF_8));
        return Base64.getEncoder().encodeToString(rawHmac);
    } catch (Exception e) {
        throw new RuntimeException("Error while calculating ");
    }
}

```

## Python

```

import sys
import hmac, hashlib, base64
username = sys.argv[1]
app_client_id = sys.argv[2]
key = sys.argv[3]
message = bytes(sys.argv[1]+sys.argv[2], 'utf-8')
key = bytes(sys.argv[3], 'utf-8')
secret_hash = base64.b64encode(hmac.new(key, message,
    digestmod=hashlib.sha256).digest()).decode()
print("SECRET HASH:",secret_hash)

```

## Mengonfirmasi akun pengguna tanpa memverifikasi email atau nomor telepon

Pemicu Lambda pra pendaftaran dapat digunakan untuk mengonfirmasi akun pengguna secara otomatis saat mendaftar, tanpa memerlukan kode konfirmasi atau memverifikasi email atau nomor telepon. Pengguna yang dikonfirmasi dengan cara ini dapat langsung masuk tanpa harus menerima kode.

Anda juga dapat menandai email atau nomor telepon pengguna yang diverifikasi melalui pemicu ini.

### Note

Meskipun pendekatan ini nyaman bagi pengguna saat mereka memulai, kami merekomendasikan verifikasi otomatis setidaknya satu dari email atau nomor telepon.

Jika tidak, pengguna dapat terbiarkan tidak dapat memulihkan jika mereka lupa kata sandi mereka.

Jika Anda tidak mengharuskan pengguna untuk menerima dan memasukkan kode konfirmasi saat mendaftar dan Anda tidak memverifikasi email dan nomor telepon secara otomatis di pemicu Lambda sebelum pendaftaran, Anda berisiko tidak memiliki alamat email atau nomor telepon terverifikasi untuk akun pengguna tersebut. Pengguna dapat memverifikasi alamat email atau nomor telepon di lain waktu. Namun, jika pengguna lupa kata sandinya dan tidak memiliki alamat email atau nomor telepon yang diverifikasi, pengguna dikunci dari akun, karena alur lupa kata sandi memerlukan email atau nomor telepon yang diverifikasi untuk mengirim kode verifikasi kepada pengguna.

## Memverifikasi kapan pengguna mengubah email atau nomor telepon mereka

Saat pengguna memperbarui alamat email atau nomor teleponnya di aplikasi Anda, Amazon Cognito segera mengirimkan pesan dengan kode verifikasi ke pengguna jika Anda mengonfigurasi kumpulan pengguna untuk memverifikasi atribut tersebut secara otomatis. Pengguna kemudian harus memberikan kode dari pesan verifikasi ke aplikasi Anda. Aplikasi Anda kemudian mengirimkan kode dalam permintaan [VerifyUserAttribute](#) API untuk menyelesaikan verifikasi nilai atribut baru.

Jika kumpulan pengguna Anda tidak mengharuskan pengguna memverifikasi alamat email atau nomor telepon yang diperbarui, Amazon Cognito segera mengubah nilai `phone_number` atribut `email` atau yang diperbarui dan menandai atribut sebagai belum diverifikasi. Pengguna Anda tidak dapat masuk dengan email atau nomor telepon yang belum diverifikasi. Mereka harus menyelesaikan verifikasi nilai yang diperbarui sebelum mereka dapat menggunakan atribut itu sebagai alias masuk.

Jika kumpulan pengguna Anda mengharuskan pengguna memverifikasi alamat email atau nomor telepon yang diperbarui, Amazon Cognito menyimpan atribut tersebut diverifikasi dan disetel ke nilai aslinya hingga pengguna Anda memverifikasi nilai atribut baru. Jika atribut adalah alias untuk login, pengguna Anda dapat masuk dengan nilai atribut asli hingga verifikasi mengubah atribut ke nilai baru. Untuk informasi selengkapnya tentang cara mengonfigurasi kumpulan pengguna agar pengguna mengharuskan pengguna memverifikasi atribut yang diperbarui, lihat [Mengonfigurasi verifikasi email atau telepon](#).

Anda dapat menggunakan pemicu Lambda pesan khusus untuk menyesuaikan pesan verifikasi. Untuk informasi selengkapnya, lihat [Pesan khusus Lambda pemicu](#). Jika alamat email atau nomor telepon pengguna tidak diverifikasi, aplikasi Anda harus memberi tahu pengguna bahwa mereka harus memverifikasi atribut, dan memberikan tombol atau tautan bagi pengguna untuk memverifikasi alamat email atau nomor telepon baru mereka.

## Proses konfirmasi dan verifikasi untuk akun pengguna yang dibuat oleh administrator atau pengembang

Akun pengguna yang dibuat oleh administrator atau developer sudah dalam status terkonfirmasi, sehingga pengguna tidak perlu memasukkan kode konfirmasi. Pesan undangan yang dikirimkan layanan Amazon Cognito kepada pengguna ini mencakup nama pengguna dan kata sandi sementara. Pengguna diharuskan untuk mengubah kata sandi sebelum masuk. Untuk informasi lebih lanjut, lihat [Sesuaikan pesan email dan SMS](#) di [Membuat akun pengguna sebagai administrator](#) dan pemicu Pesan Kustom di [Menyesuaikan alur kerja kumpulan pengguna dengan pemicu Lambda](#).

## Proses konfirmasi dan verifikasi untuk akun pengguna yang diimpor

Akun pengguna yang dibuat dengan menggunakan fitur impor pengguna di AWS Management Console, CLI, atau API (lihat [Mengimpor pengguna ke kumpulan pengguna dari file CSV](#)) sudah dalam status terkonfirmasi, sehingga pengguna tidak diharuskan memasukkan kode konfirmasi. Tidak ada pesan undangan yang dikirim. Namun, akun pengguna yang diimpor mengharuskan pengguna untuk meminta kode terlebih dahulu dengan memanggil API `ForgotPassword` dan kemudian membuat kata sandi menggunakan kode yang dikirimkan dengan memanggil API `ConfirmForgotPassword` sebelum mereka masuk. Untuk informasi lebih lanjut, lihat [Mengharuskan pengguna yang diimpor untuk mengatur ulang kata sandi mereka](#).

Email atau nomor telepon pengguna harus ditandai sebagai terverifikasi saat akun pengguna diimpor, jadi tidak diperlukan verifikasi saat pengguna masuk.

## Mengirim email saat menguji aplikasi

Amazon Cognito mengirimkan pesan email ke pengguna Anda saat mereka membuat dan mengelola akun mereka di aplikasi klien untuk kumpulan pengguna Anda. Jika Anda mengonfigurasi kolom pengguna Anda agar memerlukan verifikasi email, Amazon Cognito akan mengirimkan email saat:

- Pengguna mendaftar.
- Pengguna memperbarui alamat email mereka.
- Pengguna melakukan tindakan yang memanggil tindakan API `ForgotPassword`.
- Anda membuat akun pengguna sebagai administrator.

Tergantung pada tindakan yang memulai email, email berisi kode verifikasi atau kata sandi sementara. Pengguna Anda harus menerima email ini dan memahami pesannya. Jika tidak, mereka mungkin tidak dapat masuk dan menggunakan aplikasi Anda.

Untuk memastikan bahwa email berhasil dikirim dan pesan terlihat benar, uji tindakan di aplikasi Anda yang memulai pengiriman email dari Amazon Cognito. Sebagai contoh, dengan menggunakan halaman pendaftaran di aplikasi Anda, atau dengan menggunakan tindakan API `SignUp`, Anda dapat memulai email dengan mendaftar dengan alamat email percobaan. Saat Anda menguji dengan cara ini, ingat hal berikut:

### Penting

Saat Anda menggunakan alamat email untuk menguji tindakan yang memulai email dari Amazon Cognito, jangan gunakan alamat email palsu (yang tidak memiliki kotak pesan). Gunakan alamat email asli yang akan menerima email dari Amazon Cognito tanpa membuat pantulan keras.

Pantulan keras terjadi saat Amazon Cognito gagal mengirim email ke kotak pesan penerima, yang selalu terjadi jika kotak pesan tidak ada.

Amazon Cognito membatasi jumlah email yang dapat dikirim oleh AWS akun yang terus-menerus menimbulkan pantulan keras.

Ketika Anda menguji tindakan yang memulai email, gunakan salah satu alamat email berikut untuk mencegah pantulan keras:

- Alamat untuk akun email yang Anda miliki dan gunakan untuk pengujian. Bila Anda menggunakan alamat email Anda sendiri, Anda menerima email yang dikirim Amazon Cognito. Dengan email ini, Anda dapat menggunakan kode verifikasi untuk menguji pengalaman pendaftaran di dalam aplikasi Anda. Jika Anda menyesuaikan pesan email untuk kolam pengguna Anda, Anda dapat memeriksa apakah penyesuaian Anda terlihat sesuai.
- Alamat simulator kotak surat, `success@simulator.amazonses.com`. Jika Anda menggunakan alamat simulator, Amazon Cognito mengirimkan email dengan sukses, tetapi Anda tidak dapat melihatnya. Opsi ini berguna ketika Anda tidak perlu menggunakan kode verifikasi dan Anda tidak perlu memeriksa pesan email.
- Alamat simulator kotak surat dengan tambahan label arbitrer, seperti `success+user1@simulator.amazonses.com` atau `success+user2@simulator.amazonses.com`. Amazon Cognito berhasil mengirim email ke alamat-alamat ini, tetapi Anda tidak dapat melihat email

yang dikirimkannya. Opsi ini berguna saat Anda ingin menguji proses pendaftaran dengan menambahkan beberapa pengguna uji ke kolam pengguna Anda, dan setiap pengguna uji memiliki alamat email unik.

## Mengkonfigurasi verifikasi email atau telepon

Anda dapat memilih pengaturan untuk verifikasi email atau telepon di bawah tab Pesan. Untuk informasi selengkapnya tentang otentikasi multi-faktor (MFA), lihat MFA Pesan [Teks SMS](#).

Amazon Cognito menggunakan Amazon SNS untuk mengirim pesan SMS. Jika Anda belum mengirim pesan SMS dari Amazon Cognito atau lainnya Layanan AWS sebelumnya, Amazon SNS mungkin menempatkan akun Anda di kotak pasir SMS. Kami menyarankan Anda mengirim pesan uji ke nomor telepon terverifikasi sebelum Anda menghapus akun Anda dari kotak pasir ke produksi. Selain itu, jika Anda berencana untuk mengirim pesan SMS ke nomor telepon tujuan AS, Anda harus mendapatkan originasi atau ID Pengirim dari Amazon Pinpoint. Untuk mengonfigurasi kumpulan pengguna Amazon Cognito Anda untuk pesan SMS, lihat [Pengaturan pesan SMS untuk kolam pengguna Amazon Cognito](#)

Amazon Cognito dapat secara otomatis memverifikasi alamat email atau nomor telepon. Untuk melakukan verifikasi ini, Amazon Cognito mengirimkan kode verifikasi atau tautan verifikasi. Untuk alamat email, Amazon Cognito dapat mengirim kode atau tautan dalam pesan email. Anda dapat memilih jenis Kode atau Tautan Verifikasi saat mengedit templat pesan Verifikasi di tab Pesan di konsol Amazon Cognito. Untuk informasi selengkapnya, lihat [Menyesuaikan pesan verifikasi email](#).

Untuk nomor telepon, Amazon Cognito mengirimkan kode dalam pesan teks SMS.

Amazon Cognito harus memverifikasi nomor telepon atau alamat email untuk mengonfirmasi pengguna dan membantu mereka memulihkan kata sandi yang terlupakan. Atau, Anda dapat secara otomatis mengonfirmasi pengguna dengan pemicu Lambda pra-pendaftaran atau menggunakan operasi API [AdminConfirmSignUp](#). Untuk informasi selengkapnya, lihat [Mendaftar dan mengonfirmasi akun pengguna](#).

Kode atau tautan verifikasi berlaku selama 24 jam.

Jika Anda memilih untuk meminta verifikasi untuk alamat email atau nomor telepon, Amazon Cognito secara otomatis mengirimkan kode verifikasi atau tautan saat pengguna mendaftar. Jika kumpulan pengguna memiliki [Pemicu Lambda pengirim SMS kustom](#) atau [Pemicu Lambda pengirim email kustom](#) dikonfigurasi, fungsi tersebut dipanggil sebagai gantinya.

### Catatan

- Amazon SNS mengenakan biaya secara terpisah untuk pesan teks SMS yang digunakannya untuk memverifikasi nomor telepon. Tidak ada biaya untuk mengirim pesan email. Untuk informasi tentang harga Amazon SNS, lihat harga [SMS Seluruh Dunia](#). Untuk daftar negara saat ini di mana pesan SMS tersedia, lihat [Wilayah dan negara yang didukung](#).
- Saat Anda menguji tindakan di aplikasi yang menghasilkan pesan email dari Amazon Cognito, gunakan alamat email asli yang dapat dijangkau Amazon Cognito tanpa pantulan keras. Untuk informasi selengkapnya, lihat [the section called “Mengirim email saat menguji aplikasi”](#).
- Alur kata sandi yang terlupakan memerlukan email pengguna atau nomor telepon pengguna untuk memverifikasi pengguna.

### Important

Jika pengguna mendaftar dengan nomor telepon dan alamat email, dan pengaturan kumpulan pengguna Anda memerlukan verifikasi kedua atribut, Amazon Cognito mengirimkan kode verifikasi ke nomor telepon melalui pesan SMS. Amazon Cognito belum memverifikasi alamat email, jadi aplikasi Anda harus menelepon [GetUser](#) untuk melihat apakah alamat email menunggu verifikasi. Jika memang memerlukan verifikasi, aplikasi harus menelepon [GetUserAttributeVerificationCode](#) untuk memulai alur verifikasi email. Maka harus mengirimkan kode verifikasi dengan menelepon [VerifyUserAttribute](#).

Anda dapat menyesuaikan kuota belanja pesan SMS Anda untuk Akun AWS dan untuk pesan individual. Batas hanya berlaku untuk biaya untuk mengirim pesan SMS. Untuk informasi selengkapnya, lihat Apa itu kuota pengeluaran tingkat akun dan tingkat pesan dan bagaimana cara kerjanya? di [FAQ Amazon SNS](#).

Amazon Cognito mengirimkan pesan SMS menggunakan sumber daya Amazon SNS baik Wilayah AWS di tempat Anda membuat kumpulan pengguna atau di Wilayah alternatif Amazon SNS Legacy dari tabel berikut. Pengecualiannya adalah kumpulan pengguna Amazon Cognito di Wilayah Asia Pasifik (Seoul). Kumpulan pengguna ini menggunakan konfigurasi Amazon SNS Anda di Wilayah

Asia Pasifik (Tokyo). Untuk informasi selengkapnya, lihat [Pilih pesan Wilayah AWS SMS Amazon SNS untuk](#).

Wilayah Amazon Cognito	Legacy Amazon SNS Wilayah alternatif
AS Timur (Ohio)	US East (N. Virginia)
Asia Pasifik (Mumbai)	Asia Pasifik (Singapura)
Asia Pasifik (Seoul)	Asia Pacific (Tokyo)
(Canada (Central)	US East (N. Virginia)
Europe (Frankfurt)	Eropa (Irlandia)
Europe (London)	Eropa (Irlandia)

Contoh: Jika kumpulan pengguna Amazon Cognito Anda berada di Asia Pasifik (Mumbai), dan Anda telah meningkatkan batas pengeluaran di ap-southeast-1, Anda mungkin tidak ingin meminta peningkatan terpisah di ap-south-1. Sebagai gantinya, Anda dapat menggunakan sumber daya Amazon SNS Anda di Asia Pasifik (Singapura).

### Memverifikasi pembaruan alamat email dan nomor telepon

Alamat email atau atribut nomor telepon dapat menjadi aktif dan tidak diverifikasi segera setelah pengguna Anda mengubah nilainya. Amazon Cognito juga dapat mengharuskan pengguna Anda memverifikasi nilai baru sebelum Amazon Cognito memperbarui atribut. Saat Anda mengharuskan pengguna memverifikasi nilai baru terlebih dahulu, mereka dapat menggunakan nilai asli untuk masuk dan menerima pesan hingga mereka memverifikasi nilai baru.

Bila pengguna Anda dapat menggunakan alamat email atau nomor telepon mereka sebagai alias masuk di kumpulan pengguna Anda, nama login mereka untuk atribut yang diperbarui bergantung pada apakah Anda memerlukan verifikasi atribut yang diperbarui. Bila Anda mengharuskan pengguna memverifikasi atribut yang diperbarui, pengguna dapat masuk dengan nilai atribut asli hingga mereka memverifikasi nilai baru. Jika Anda tidak mengharuskan pengguna memverifikasi atribut yang diperbarui, pengguna tidak dapat masuk atau menerima pesan di nilai atribut baru atau asli hingga mereka memverifikasi nilai baru.

Misalnya, kumpulan pengguna Anda mengizinkan login dengan alias alamat email, dan mengharuskan pengguna memverifikasi alamat email mereka saat mereka memperbarui. Sue, yang masuk sebagai `sue@example.com`, ingin mengubah alamat emailnya `sue2@example.com` tetapi secara tidak sengaja masuk `sue2@example.com`. Sue tidak menerima email verifikasi, jadi dia tidak bisa memverifikasi `sue2@example.com`. Sue masuk sebagai `sue@example.com` dan mengirimkan kembali formulir di aplikasi Anda untuk memperbarui alamat emailnya `sue2@example.com`. Dia menerima email ini, memberikan kode verifikasi ke aplikasi Anda, dan mulai masuk sebagai `sue2@example.com`.

Saat pengguna memperbarui atribut dan kumpulan pengguna Anda memverifikasi nilai atribut baru

- Mereka dapat masuk dengan nilai atribut asli sebelum mereka mengkonfirmasi kode untuk memverifikasi nilai baru.
- Mereka hanya dapat masuk dengan nilai atribut baru setelah mereka mengkonfirmasi kode untuk memverifikasi nilai baru.
- Jika Anda menyetel `email_verified` atau `phone_number_verified` masuk `true` dalam permintaan [AdminUpdateUserAttributes](#) API, mereka dapat masuk sebelum mengonfirmasi kode yang dikirimkan Amazon Cognito kepada mereka.

Saat pengguna memperbarui atribut dan kumpulan pengguna Anda tidak memverifikasi nilai atribut baru

- Mereka tidak dapat masuk dengan, atau menerima pesan di, nilai atribut asli.
- Mereka tidak dapat masuk dengan, atau menerima pesan selain kode konfirmasi di, nilai atribut baru sebelum mereka mengonfirmasi kode untuk memverifikasi nilai baru.
- Jika Anda menyetel `email_verified` atau `phone_number_verified` masuk `true` dalam permintaan [AdminUpdateUserAttributes](#) API, mereka dapat masuk sebelum mengonfirmasi kode yang dikirimkan Amazon Cognito kepada mereka.

Untuk mewajibkan verifikasi atribut saat pengguna memperbarui alamat email atau nomor telepon mereka

1. Masuk ke [konsol Amazon Cognito](#). Jika diminta, masukkan AWS kredensial Anda.
2. Di panel navigasi, pilih Kumpulan Pengguna, dan pilih kumpulan pengguna yang ingin Anda edit.
3. Di tab Pengalaman pendaftaran, pilih Edit di bawah Verifikasi atribut dan konfirmasi akun pengguna.



4. Pilih Pertahankan nilai atribut asli tetap aktif saat pembaruan tertunda.
5. Di bawah Nilai atribut Aktif saat pembaruan tertunda, pilih atribut yang ingin diverifikasi pengguna sebelum Amazon Cognito memperbarui nilainya.
6. Pilih Simpan perubahan.

Untuk meminta verifikasi pembaruan atribut dengan Amazon Cognito API, Anda dapat mengatur `AttributesRequireVerificationBeforeUpdate` parameter dalam permintaan.

[UpdateUserPool](#)

Mengotorisasi Amazon Cognito untuk mengirim pesan SMS atas nama Anda

Untuk mengirim pesan SMS ke pengguna atas nama Anda, Amazon Cognito memerlukan izin Anda. Untuk memberikan izin tersebut, Anda dapat membuat peran AWS Identity and Access Management (IAM). Di tab Pesan konsol Amazon Cognito di bawah SMS, pilih Edit untuk menetapkan peran.

Mengkonfigurasi pesan verifikasi SMS dan email dan pesan undangan pengguna

Amazon Cognito memungkinkan Anda menyesuaikan pesan verifikasi SMS dan email, serta pesan undangan pengguna, untuk meningkatkan keamanan dan pengalaman pengguna aplikasi Anda. Dengan Amazon Cognito, Anda dapat memilih antara verifikasi tautan berbasis kode atau satu-klik yang sesuai dengan kebutuhan aplikasi Anda. Topik ini membahas bagaimana Anda dapat mempersonalisasi otentikasi multi-faktor (MFA) dan komunikasi verifikasi di konsol Amazon Cognito.

Di tab Pesan di bawah Templat pesan, Anda dapat menyesuaikan:

- Pesan SMS Anda multi-faktor otentikasi (MFA) pesan teks
- Pesan verifikasi SMS dan email Anda
- Jenis verifikasi untuk email—kode atau tautan
- Pesan undangan pengguna Anda
- DARI dan BALAS-KE alamat email untuk email melalui kumpulan pengguna Anda

#### Note

Templat pesan verifikasi SMS dan email hanya muncul jika Anda telah memilih untuk mewajibkan verifikasi nomor telepon dan email di tab Verifikasi. Demikian pula, template pesan SMS MFA hanya muncul jika pengaturan MFA diperlukan atau opsional.

## Topik

- [Template pesan](#)
- [Menyesuaikan pesan SMS](#)
- [Menyesuaikan pesan verifikasi email](#)
- [Menyesuaikan pesan undangan pengguna](#)
- [Menyesuaikan alamat email Anda](#)
- [Mengotorisasi Amazon Cognito untuk mengirim email Amazon SES atas nama Anda \(dari alamat email FROM khusus\)](#)

## Template pesan

Anda dapat menggunakan templat pesan untuk menyisipkan bidang ke dalam pesan menggunakan placeholder yang diganti oleh nilai yang sesuai.

## Placeholder templat

Deskripsi	Token
Kode verifikasi	{#####}
Kata sandi sementara	{#####}
Nama pengguna	{username}

### Note

Anda tidak dapat menggunakan {username} placeholder dalam pesan email verifikasi. Anda dapat menggunakan {username} placeholder dalam pesan email undangan yang Anda hasilkan dengan operasi. [AdminCreateUser](#) Pesan email undangan ini menggunakan dua placeholder: nama pengguna, as{username}, dan kata sandi sementara, sebagai {#####}

Anda dapat menggunakan placeholder template keamanan tingkat lanjut untuk melakukan hal berikut:

- Sertakan detail spesifik tentang acara seperti alamat IP, kota, negara, waktu masuk, dan nama perangkat. Fitur keamanan canggih Amazon Cognito dapat menganalisis detail ini.
- Verifikasi apakah tautan sekali klik valid.
- Gunakan ID peristiwa, token umpan balik, dan nama pengguna untuk membuat tautan satu klik Anda sendiri.

#### Note

Untuk menghasilkan tautan sekali klik dan menggunakan `{one-click-link-valid}` dan `{one-click-link-invalid}` placeholder di templat email keamanan tingkat lanjut, Anda harus sudah memiliki domain yang dikonfigurasi untuk kumpulan pengguna Anda.

#### Placeholder templat keamanan tingkat lanjut

Deskripsi	Token
Alamat IP	<code>{ip-address}</code>
Kota	<code>{city}</code>
Negara	<code>{country}</code>
Waktu masuk	<code>{login-time}</code>
Nama perangkat	<code>{device-name}</code>
Satu-klik link valid	<code>{one-click-link-valid}</code>
Tautan sekali klik tidak valid	<code>{one-click-link-invalid}</code>
ID peristiwa	<code>{event-id}</code>
Token umpan balik	<code>{feedback-token}</code>

## Menyesuaikan pesan SMS

### Note

Dalam pengalaman konsol Amazon Cognito yang baru, Anda dapat menyesuaikan pesan SMS

Anda dapat menyesuaikan pesan SMS untuk otentikasi multi-faktor (MFA) di tab Pesan di bawah judul Templat pesan.

### Important

Pesan kustom Anda harus berisi {####} placeholder. Placeholder ini diganti dengan kode otentikasi sebelum pesan dikirim.

Amazon Cognito memaksakan panjang maksimum untuk pesan SMS, termasuk kode otentikasi, pada 140 karakter UTF-8.

## Menyesuaikan pesan verifikasi SMS

Anda dapat menyesuaikan pesan SMS untuk verifikasi nomor telepon dengan mengedit templat pada heading Apakah Anda ingin menyesuaikan pesan verifikasi SMS Anda?.

### Important

Pesan kustom Anda harus berisi {####} placeholder. Placeholder ini diganti dengan kode verifikasi sebelum pesan dikirim.

Panjang maksimum untuk pesan, termasuk kode verifikasi, adalah 140 karakter UTF-8.

## Menyesuaikan pesan verifikasi email

Untuk memverifikasi alamat email pengguna di kumpulan pengguna Anda dengan Amazon Cognito, Anda dapat mengirim pesan email kepada pengguna dengan tautan yang dapat mereka pilih, atau Anda dapat mengirimi mereka kode yang dapat mereka masukkan.

Untuk menyesuaikan subjek email dan konten pesan untuk pesan verifikasi alamat email, edit templat pesan Verifikasi di tab Pesan kumpulan pengguna Anda. Anda dapat memilih jenis Kode atau Tautan Verifikasi saat mengedit templat pesan Verifikasi.

Bila Anda memilih Kode sebagai jenis verifikasi, pesan kustom Anda harus berisi {####} placeholder. Saat Anda mengirim pesan, kode verifikasi menggantikan placeholder ini.

Saat Anda memilih Tautan sebagai jenis verifikasi, pesan khusus Anda harus menyertakan placeholder dalam format. {##Verify Your Email##} Anda dapat mengubah string teks antara karakter placeholder, misalnya. {##Click here##} Tautan verifikasi berjudul Verifikasi Email Anda menggantikan placeholder ini.

Tautan untuk pesan verifikasi email mengarahkan pengguna Anda ke URL seperti contoh berikut.

```
https://<your user pool domain>/confirmUser/?  
client_id=abcdefg12345678&user_name=emailtest&confirmation_code=123456
```

Panjang maksimum pesan, termasuk kode verifikasi (jika ada), adalah 20.000 karakter UTF-8. Anda dapat menggunakan tag HTML dalam pesan ini untuk memformat konten.

### Menyesuaikan pesan undangan pengguna

Anda dapat menyesuaikan pesan undangan pengguna yang dikirimkan Amazon Cognito ke pengguna baru melalui SMS atau pesan email dengan mengedit templat pesan Undangan di tab Pesan.

#### Important

Pesan kustom Anda harus berisi {username} dan {####} placeholder. Saat Amazon Cognito mengirim pesan undangan, ia menggantikan placeholder ini dengan nama pengguna dan kata sandi pengguna Anda.

Panjang maksimum pesan SMS, termasuk kode verifikasi, adalah 140 karakter UTF-8. Panjang maksimum pesan email, termasuk kode verifikasi, adalah 20.000 karakter UTF-8. Anda dapat menggunakan tag HTML dalam pesan email Anda untuk memformat konten.

## Menyesuaikan alamat email Anda

Secara default, Amazon Cognito mengirimkan pesan email ke pengguna di kumpulan pengguna Anda dari alamat `no-reply@verificationemail.com`. Anda dapat memilih untuk menentukan alamat email FROM dan REPLY-TO kustom, bukan `no-reply@verificationemail.com`.

Untuk menyesuaikan alamat email FROM dan REPLY-TO

1. Arahkan ke [konsol Amazon Cognito](#), dan pilih Kumpulan Pengguna.
2. Pilih kolam pengguna yang ada dari daftar, atau [buat kolam pengguna](#).
3. Pilih tab Pesan. Di bawah Email, pilih Edit.
4. Pilih Wilayah SES.
5. Pilih alamat email FROM dari daftar alamat email yang telah Anda verifikasi dengan Amazon SES di Wilayah SES yang Anda pilih. Untuk menggunakan alamat email dari domain terverifikasi, konfigurasi setelan email di AWS Command Line Interface atau AWS API. Untuk informasi selengkapnya, lihat [Memverifikasi alamat email dan domain di Amazon SES](#) di Panduan Pengembang Layanan Email Sederhana Amazon.
6. Pilih set Konfigurasi dari daftar set konfigurasi di Wilayah SES yang Anda pilih.
7. Masukkan nama pengirim FROM yang ramah untuk pesan email Anda, dalam format `John Stiles <johnstiles@example.com>`.
8. Untuk menyesuaikan alamat email BALAS-KE, masukkan alamat email yang valid di bidang alamat email BALAS-KE.

Mengotorisasi Amazon Cognito untuk mengirim email Amazon SES atas nama Anda (dari alamat email FROM khusus)

Anda dapat mengonfigurasi Amazon Cognito untuk mengirim email dari alamat email FROM kustom alih-alih alamat defaultnya. Untuk menggunakan alamat khusus, Anda harus memberikan izin Amazon Cognito untuk mengirim pesan email dari identitas terverifikasi Amazon SES. Dalam kebanyakan kasus, Anda dapat memberikan izin dengan membuat kebijakan otorisasi pengiriman. Untuk informasi selengkapnya, lihat [Menggunakan otorisasi pengiriman dengan Amazon SES](#) di Panduan Pengembang Layanan Email Sederhana Amazon.

Saat Anda mengonfigurasi kumpulan pengguna untuk menggunakan Amazon SES untuk pesan email, Amazon Cognito membuat `AWSServiceRoleForAmazonCognitoIdpEmailService` peran di akun Anda untuk memberikan akses ke Amazon SES. Tidak diperlukan kebijakan otorisasi

pengiriman saat peran `AWSServiceRoleForAmazonCognitoIdpEmailService` terkait layanan digunakan. Anda hanya perlu menambahkan kebijakan otorisasi pengiriman saat Anda menggunakan fungsionalitas email default di kumpulan pengguna dan identitas Amazon SES yang diverifikasi sebagai alamat FROM.

Untuk informasi lebih lanjut tentang peran yang terkait dengan layanan yang dibuat Amazon Cognito, lihat [Menggunakan peran terkait layanan untuk Amazon Cognito](#).

Contoh berikut mengirim kebijakan otorisasi memberi Amazon Cognito kemampuan terbatas untuk menggunakan identitas terverifikasi Amazon SES. Amazon Cognito hanya dapat mengirim pesan email ketika melakukannya atas nama kumpulan pengguna dalam `aws:SourceArn` kondisi dan akun dalam kondisi tersebut `aws:SourceAccount`. Untuk contoh selengkapnya, lihat [Amazon SES mengirimkan contoh kebijakan otorisasi](#) di Panduan Pengembang Layanan Email Sederhana Amazon.

#### Note

Dalam contoh ini, nilai "Sid" adalah string arbitrer yang secara unik mengidentifikasi pernyataan. Untuk informasi selengkapnya tentang sintaks kebijakan, lihat [Amazon SES mengirim kebijakan otorisasi](#) di Panduan Pengembang Layanan Email Sederhana Amazon.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "stmnt1234567891234",
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "email.cognito-idp.amazonaws.com"
        ]
      },
      "Action": [
        "SES:SendEmail",
        "SES:SendRawEmail"
      ],
      "Resource": "<your SES identity ARN>",
      "Condition": {
        "StringEquals": {
```

```
        "aws:SourceAccount": "<your account number>"
    },
    "ArnLike": {
        "aws:SourceArn": "<your user pool ARN>"
    }
}
]
```

Konsol Amazon Cognito menambahkan kebijakan serupa untuk Anda saat Anda memilih identitas Amazon SES dari menu tarik-turun. Jika Anda menggunakan CLI atau API untuk mengonfigurasi kumpulan pengguna, Anda harus melampirkan kebijakan yang terstruktur seperti contoh sebelumnya ke Identitas Amazon SES Anda.

## Membuat akun pengguna sebagai administrator

Setelah Anda membuat kolam pengguna Anda, Anda dapat membuat pengguna menggunakan AWS Management Console, serta AWS Command Line Interface atau API Amazon Cognito. Anda dapat membuat profil untuk pengguna baru di kolam pengguna dan mengirim pesan selamat datang dengan petunjuk pendaftaran kepada pengguna melalui SMS atau email.

Developer dan administrator dapat melakukan tugas berikut:

- Membuat profil pengguna baru dengan menggunakan AWS Management Console atau dengan memanggil API `AdminCreateUser`.
- Tetapkan nilai atribut pengguna.
- Buat atribut kustom.
- Tetapkan nilai atribut kustom yang tidak dapat diubah dalam permintaan `AdminCreateUser` API. Fitur ini tidak tersedia di konsol Amazon Cognito.
- Tentukan kata sandi sementara atau mengizinkan Amazon Cognito untuk secara otomatis membuatnya.
- Tentukan apakah alamat email dan nomor telepon yang diberikan ditandai sebagai terverifikasi untuk pengguna baru.
- Tentukan pesan undangan SMS dan email khusus untuk pengguna baru melalui AWS Management Console atau pemicu Pesan Kustom Lambda. Untuk informasi lebih lanjut, lihat [Menyesuaikan alur kerja kumpulan pengguna dengan pemicu Lambda](#).



- Tentukan apakah pesan undangan dikirim melalui SMS, email, atau keduanya.
- Kirim ulang pesan selamat datang ke pengguna yang sudah ada dengan memanggil API `AdminCreateUser`, menentukan `RESEND` untuk parameter `MessageAction`.

#### Note

Tindakan ini saat ini tidak dapat dilakukan menggunakan AWS Management Console.

- Tekan pengiriman pesan undangan saat pengguna dibuat.
- Tentukan batas waktu kedaluwarsa untuk akun pengguna (sampai 90 hari).
- Izinkan pengguna untuk mendaftar sendiri atau haruskan pengguna baru ditambahkan hanya oleh administrator.

## Alur otentikasi untuk pengguna yang dibuat oleh administrator atau pengembang

Alur autentikasi untuk para pengguna ini termasuk langkah lebih untuk memasukkan kata sandi baru dan memberikan apapun nilai yang hilang untuk atribut yang diperlukan. Langkah-langkahnya diuraikan berikutnya; langkah 5, 6, dan 7 khusus untuk para pengguna ini.

1. Pengguna mulai masuk untuk pertama kalinya dengan mengirimkan nama pengguna dan kata sandi mereka.
2. SDK memanggil `InitiateAuth(Username, USER_SRP_AUTH)`.
3. Amazon Cognito mengembalikan `PASSWORD_VERIFIER` tantangan dengan blok Salt & Secret.
4. SDK melakukan penghitungan SRP dan memanggil `RespondToAuthChallenge(Username, <SRP variables>, PASSWORD_VERIFIER)`.
5. Amazon Cognito mengembalikan tantangannya. `NEW_PASSWORD_REQUIRED` Isi tantangan ini mencakup atribut pengguna saat ini, dan atribut apa pun yang diperlukan di kumpulan pengguna Anda yang saat ini tidak memiliki nilai di profil pengguna. Untuk informasi lebih lanjut, lihat [RespondToAuthChallenge](#).
6. Pengguna diminta dan memasukkan kata sandi baru dan nilai yang hilang untuk atribut yang diperlukan.
7. SDK memanggil `RespondToAuthChallenge(Username, <New password>, <User attributes>)`.
8. Jika pengguna memerlukan faktor kedua untuk MFA, Amazon Cognito mengembalikan tantangan `SMS_MFA` dan kodenya dikirimkan.

9. Setelah pengguna berhasil mengubah kata sandinya dan secara opsional memberikan nilai yang terkait atau menyelesaikan MFA, pengguna masuk dan token diterbitkan.

Ketika pengguna telah memenuhi semua tantangan, layanan Amazon Cognito menandai pengguna sebagai terkonfirmasi dan menerbitkan ID, akses, dan menyegarkan token untuk pengguna. Untuk informasi lebih lanjut, lihat [Menggunakan token dengan kolam pengguna](#).

## Membuat pengguna baru di AWS Management Console

Anda dapat mengatur persyaratan kata sandi pengguna, mengonfigurasi pesan undangan dan verifikasi yang dikirim ke pengguna, dan menambahkan pengguna baru dengan konsol Amazon Cognito.

Tetapkan kebijakan kata sandi dan aktifkan pendaftaran mandiri

Anda dapat mengonfigurasi pengaturan untuk kompleksitas kata sandi minimum dan apakah pengguna dapat mendaftar menggunakan API publik di kumpulan pengguna Anda.


Konfigurasi kebijakan kata sandi

1. Arahkan ke [konsol Amazon Cognito](#), dan pilih Kumpulan Pengguna.
2. Pilih kolam pengguna yang ada dari daftar, atau [buat kolam pengguna](#).
3. Pilih tab Pengalaman masuk dan temukan kebijakan Kata Sandi. Pilih Edit.
4. Pilih mode kebijakan Kata Sandi Kustom.
5. Pilih panjang minimum Kata Sandi. Untuk batas persyaratan panjang kata sandi, lihat [Kuota sumber daya kumpulan pengguna](#).
6. Pilih persyaratan kompleksitas Kata Sandi.
7. Pilih berapa lama kata sandi yang ditetapkan oleh administrator harus valid.
8. Pilih Simpan perubahan.

Izinkan pendaftaran swalayan

1. Arahkan ke [konsol Amazon Cognito](#), dan pilih Kumpulan Pengguna.
2. Pilih kolam pengguna yang ada dari daftar, atau [buat kolam pengguna](#).
3. Pilih tab Sign-up experience dan temukan Self-service sign-up. Pilih Edit.

4. Pilih apakah akan Aktifkan pendaftaran mandiri. Registrasi mandiri biasanya digunakan dengan klien aplikasi publik yang perlu mendaftarkan pengguna baru di kumpulan pengguna Anda tanpa mendistribusikan rahasia klien atau kredensi API AWS Identity and Access Management (IAM).

 Menonaktifkan registrasi mandiri


Jika Anda tidak mengaktifkan registrasi mandiri, pengguna baru harus dibuat dengan tindakan API administratif menggunakan kredensial API IAM atau dengan login dengan penyedia federasi.

5. Pilih Simpan perubahan.

Sesuaikan pesan email dan SMS

Sesuaikan pesan pengguna

Anda dapat menyesuaikan pesan yang dikirimkan Amazon Cognito kepada pengguna Anda saat Anda mengundang mereka untuk masuk, mereka mendaftar untuk akun pengguna, atau mereka masuk dan diminta untuk otentikasi multi-faktor (MFA).

 Note

Pesan Undangan dikirim saat Anda membuat pengguna di kumpulan pengguna dan mengundang mereka untuk masuk. Amazon Cognito mengirimkan informasi login awal ke alamat email atau nomor telepon pengguna.

Pesan Verifikasi dikirim saat pengguna mendaftar untuk akun pengguna di kumpulan pengguna Anda. Amazon Cognito mengirimkan kode ke pengguna. Saat pengguna memberikan kode ke Amazon Cognito, mereka memverifikasi informasi kontak mereka dan mengonfirmasi akun mereka untuk masuk. Kode verifikasi valid untuk 24 jam.

Pesan MFA dikirim saat Anda mengaktifkan SMS MFA di kumpulan pengguna Anda, dan pengguna yang telah mengonfigurasi masuk MFA SMS dan diminta untuk MFA.

1. Arahkan ke [konsol Amazon Cognito](#), dan pilih Kumpulan Pengguna.
2. Pilih kolam pengguna yang ada dari daftar, atau [buat kolam pengguna](#).
3. Pilih tab Pesan dan temukan templat Pesan. Pilih Pesan verifikasi, pesan Undangan, atau pesan MFA dan pilih Edit.

#### 4. Sesuaikan pesan untuk jenis pesan yang dipilih.

##### Note

Semua variabel dalam template pesan harus disertakan saat Anda menyesuaikan pesan. Jika variabel, misalnya {#####}, tidak disertakan, pengguna Anda tidak akan memiliki informasi yang cukup untuk menyelesaikan tindakan pesan. Untuk informasi selengkapnya, lihat [Templat pesan](#).

#### 5. a. Pesan verifikasi

- i. Pilih jenis Verifikasi untuk pesan Email. Verifikasi Kode mengirimkan kode numerik yang harus dimasukkan pengguna. Verifikasi tautan mengirimkan tautan yang dapat diklik pengguna untuk memverifikasi informasi kontak mereka. Teks dalam variabel untuk pesan Link ditampilkan sebagai teks hyperlink. Misalnya, template pesan menggunakan variabel {# #Click here ##} ditampilkan sebagai [Klik di sini di](#) pesan email.
- ii. Masukkan subjek Email untuk pesan Email.
- iii. Masukkan template pesan Email kustom untuk pesan Email. Anda dapat menyesuaikan template ini dengan HTML.
- iv. Masukkan templat pesan SMS khusus untuk pesan SMS.
- v. Pilih Simpan perubahan.

#### b. Pesan undangan

- i. Masukkan subjek Email untuk pesan Email.
- ii. Masukkan template pesan Email kustom untuk pesan Email. Anda dapat menyesuaikan template ini dengan HTML.
- iii. Masukkan templat pesan SMS khusus untuk pesan SMS.
- iv. Pilih Simpan perubahan.

#### c. Pesan MFA

- i. Masukkan templat pesan SMS khusus untuk pesan SMS.
- ii. Pilih Simpan perubahan.

## Buat pengguna

### Buat pengguna

Anda dapat membuat pengguna baru untuk kumpulan pengguna Anda dari konsol Amazon Cognito. Biasanya, pengguna dapat masuk setelah mereka menetapkan kata sandi. Untuk masuk dengan alamat email, pengguna harus memverifikasi email atribut. Untuk masuk dengan nomor telepon, pengguna harus memverifikasi phone\_number atribut. Untuk mengonfirmasi akun sebagai administrator, Anda juga dapat menggunakan AWS CLI atau API, atau membuat profil pengguna dengan penyedia identitas gabungan. Untuk informasi selengkapnya, lihat Referensi [API Amazon Cognito](#).

1. Arahkan ke [konsol Amazon Cognito](#), dan pilih Kumpulan Pengguna.
2. Pilih kolam pengguna yang ada dari daftar, atau [buat kolam pengguna](#).
3. Pilih tab Pengguna, dan pilih Buat pengguna.
4. Tinjau persyaratan masuk dan keamanan kumpulan Pengguna untuk panduan tentang persyaratan kata sandi, metode pemulihan akun yang tersedia, dan atribut alias untuk kumpulan pengguna Anda.
5. Pilih bagaimana Anda ingin mengirim pesan Undangan. Pilih pesan SMS, pesan email, atau keduanya.

#### Note

Sebelum Anda dapat mengirim pesan undangan, konfigurasi pengirim dan Wilayah AWS dengan Amazon Simple Notification Service dan Amazon Simple Email Service di tab Messaging pool pengguna Anda. Pesan penerima dan tarif data berlaku. Amazon SES menagih Anda untuk pesan email secara terpisah, dan Amazon SNS menagih Anda untuk pesan SMS secara terpisah.

6. Pilih nama pengguna untuk pengguna baru.
7. Pilih apakah Anda ingin Buat kata sandi atau minta Amazon Cognito Buat kata sandi untuk pengguna. Setiap kata sandi sementara harus mematuhi kebijakan kata sandi kumpulan pengguna.
8. Pilih Buat.
9. Pilih tab Pengguna, dan pilih entri Nama pengguna untuk pengguna. Tambahkan dan edit atribut Pengguna dan keanggotaan Grup. Tinjau riwayat acara Pengguna.

## Menambahkan grup ke kumpulan pengguna

Dukungan untuk grup di kolam pengguna Amazon Cognito memungkinkan Anda membuat dan mengelola grup, menambahkan pengguna ke grup, dan menghapus pengguna dari grup. Gunakan grup untuk membuat koleksi pengguna guna mengelola izin mereka atau untuk mewakili tipe pengguna yang berbeda. Anda dapat menetapkan peran AWS Identity and Access Management (IAM) ke grup untuk menentukan izin bagi anggota grup.

Anda dapat menggunakan grup untuk membuat kumpulan pengguna di kolam pengguna, yang sering dilakukan untuk mengatur izin bagi pengguna tersebut. Sebagai contoh, Anda dapat membuat grup terpisah untuk pengguna yang merupakan pembaca, kontributor, dan editor situs web dan aplikasi Anda. Dengan menggunakan IAM role yang terkait dengan grup, Anda juga dapat mengatur izin berbeda untuk grup berbeda tersebut sehingga hanya kontributor yang dapat memasukkan konten ke Amazon S3 dan hanya editor yang dapat menerbitkan konten melalui API di Amazon API Gateway.

Anda dapat membuat dan mengelola grup di kumpulan pengguna dari AWS Management Console, API, dan CLI. Sebagai pengembang (menggunakan AWS kredensi), Anda dapat membuat, membaca, memperbarui, menghapus, dan membuat daftar grup untuk kumpulan pengguna. Anda juga dapat menambahkan pengguna dan menghapus pengguna dari grup.

Tidak ada biaya tambahan untuk menggunakan grup dalam kolam pengguna. Lihat [Harga Amazon Cognito](#) untuk informasi selengkapnya.

## Menetapkan peran IAM ke grup

Anda dapat menggunakan grup untuk mengontrol izin ke sumber daya Anda menggunakan IAM role. IAM role mencakup kebijakan kepercayaan dan kebijakan izin. Kebijakan [kepercayaan](#) peran menentukan siapa yang dapat menggunakan peran. Kebijakan [izin](#) menentukan tindakan dan sumber daya yang dapat diakses oleh anggota grup Anda. Saat Anda membuat peran IAM, siapkan kebijakan kepercayaan peran untuk memungkinkan pengguna grup Anda mengambil peran tersebut. Dalam kebijakan izin peran, tentukan izin yang Anda inginkan untuk dimiliki grup Anda.

Saat Anda membuat grup di Amazon Cognito, Anda menentukan IAM role dengan menyediakan [ARN](#) peran. Saat anggota grup masuk menggunakan Amazon Cognito, mereka dapat menerima kredensial sementara dari kolam identitas. Izin mereka ditentukan oleh IAM role terkait.

Pengguna individu dapat berada di dalam beberapa grup. Sebagai developer, Anda memiliki opsi berikut untuk memilih IAM role secara otomatis saat pengguna berada di beberapa grup:

- Anda dapat menetapkan nilai prioritas untuk setiap grup. Grup dengan prioritas yang lebih baik (lebih rendah) akan dipilih dan IAM role terkaitnya akan diterapkan.
- Aplikasi Anda juga dapat memilih di antara peran yang tersedia saat meminta AWS kredensial untuk pengguna melalui kumpulan identitas, dengan menentukan peran ARN dalam parameter. [GetCredentialsForIdentityCustomRoleARN](#) IAM role yang ditentukan harus cocok dengan peran yang tersedia untuk pengguna.

## Menetapkan nilai prioritas ke grup

Seorang pengguna dapat menjadi anggota lebih dari satu grup. Dalam akses pengguna dan token ID, `cognito:groups` klaim berisi daftar semua grup yang dimiliki pengguna. Klaim `cognito:roles` berisi daftar peran yang sesuai dengan grup.

Karena pengguna dapat menjadi bagian dari lebih dari satu grup, setiap grup dapat diberikan prioritas. Ini adalah angka non-negatif yang menentukan prioritas grup ini relatif terhadap grup lain yang dimiliki pengguna di kumpulan pengguna. Nol adalah nilai prioritas teratas. Grup dengan nilai prioritas yang lebih rendah lebih diutamakan daripada grup dengan nilai prioritas yang lebih tinggi atau nol. Jika pengguna termasuk dalam dua grup atau lebih, grup dengan nilai prioritas terendah akan memiliki peran IAM yang diterapkan pada `cognito:preferred_role` klaim dalam token ID pengguna.

Dua grup dapat memiliki nilai prioritas yang sama. Jika ini terjadi, tidak ada grup yang didahulukan dari yang lain. Jika dua grup dengan nilai prioritas yang sama memiliki peran ARN yang sama, peran tersebut digunakan dalam klaim `cognito:preferred_role` dalam token ID untuk pengguna di setiap grup. Jika kedua grup memiliki peran ARN yang berbeda, klaim `cognito:preferred_role` tidak diatur di dalam token ID pengguna.

## Menggunakan grup untuk mengontrol izin dengan Amazon API Gateway

Anda dapat menggunakan grup di kolam pengguna untuk mengontrol izin dengan Amazon API Gateway. Grup yang menjadi anggota pengguna termasuk dalam token ID dan token akses dari kumpulan pengguna dalam `cognito:groups` klaim. Anda dapat mengirimkan ID atau token akses dengan permintaan ke Amazon API Gateway dan menggunakan otorisasi kumpulan pengguna Amazon Cognito untuk REST API. Untuk informasi selengkapnya, lihat [Mengendalikan akses ke REST API menggunakan kolam pengguna Amazon Cognito sebagai pemberi otorisasi](#) di [Panduan Developer API Gateway](#).

Anda juga dapat mengotorisasi akses ke API HTTP Amazon API Gateway dengan otorisasi JWT kustom. Untuk informasi selengkapnya, lihat [Mengendalikan akses ke API HTTP dengan otorisasi JWT](#) di [Panduan Developer API Gateway](#).

## Batasan pada kelompok

Grup pengguna tunduk pada batasan berikut:

- Jumlah grup yang dapat Anda buat dibatasi oleh kuota [layanan Amazon Cognito](#).
- Grup tidak dapat disarangkan.
- Anda tidak dapat mencari pengguna di dalam grup.
- Anda tidak dapat mencari grup berdasarkan nama, tetapi Anda dapat membuat daftar grup.

## Membuat grup baru di AWS Management Console

Gunakan prosedur berikut untuk membuat grup baru.

Untuk membuat grup baru

1. Masuk ke [Konsol Amazon Cognito](#). Jika diminta, masukkan AWS kredensial Anda.
2. Pilih Kolam Pengguna.
3. Pilih kumpulan pengguna yang ada dari daftar.
4. Pilih tab Grup, lalu pilih Buat grup.
5. Pada halaman Buat grup, di Nama grup, masukkan nama ramah untuk grup baru Anda.
6. Anda dapat secara opsional memberikan informasi tambahan tentang grup ini menggunakan salah satu bidang berikut:
  - Deskripsi - Masukkan detail tentang apa grup baru ini akan digunakan.
  - Prioritas - Amazon Cognito mengevaluasi dan menerapkan semua izin grup untuk pengguna tertentu berdasarkan grup mana mereka memiliki nilai prioritas yang lebih rendah. Kelompok dengan prioritas lebih rendah akan dipilih dan peran IAM terkait akan diterapkan. Untuk informasi selengkapnya, lihat [Menetapkan nilai prioritas ke grup](#).
  - Peran IAM - Anda dapat menetapkan peran IAM ke grup Anda ketika Anda perlu mengontrol izin ke sumber daya Anda. Jika Anda mengintegrasikan kolam pengguna dengan kolam identitas, pengaturan IAM role menentukan peran mana yang ditetapkan dalam token ID pengguna jika kolam identitas terkonfigurasi untuk memilih peran dari token. Untuk informasi selengkapnya, lihat [Menetapkan peran IAM ke grup](#).



- Tambahkan pengguna ke grup ini - Tambahkan pengguna yang ada sebagai anggota grup ini setelah dibuat.

7. Pilih Buat untuk mengonfirmasi.

## Mengelola dan mencari akun pengguna

Setelah membuat kolam pengguna, Anda dapat melihat dan mengelola pengguna menggunakan AWS Management Console, serta AWS Command Line Interface atau API Amazon Cognito.

Topik ini menjelaskan bagaimana Anda dapat melihat dan mencari pengguna menggunakan AWS Management Console.

### Melihat atribut pengguna

Gunakan prosedur berikut untuk melihat atribut pengguna di konsol Amazon Cognito.

Untuk melihat atribut pengguna

1. Masuk ke [Konsol Amazon Cognito](#). Jika diminta, masukkan AWS kredensi.
2. Pilih Kolam Pengguna.
3. Pilih kumpulan pengguna yang ada dari daftar.
4. Pilih Pengguna tab, dan kemudian pilih pengguna dalam daftar.
5. Pada halaman detail pengguna, di bawah Atribut pengguna, Anda dapat melihat atribut mana yang terkait dengan pengguna.

### Menyetel ulang kata sandi pengguna

Gunakan prosedur berikut untuk mengatur ulang kata sandi pengguna di konsol Amazon Cognito.

Untuk mengatur ulang kata sandi pengguna

1. Masuk ke [Konsol Amazon Cognito](#). Jika diminta, masukkan AWS kredensi.
2. Pilih Kolam Pengguna.
3. Pilih kumpulan pengguna yang ada dari daftar.
4. Pilih Pengguna tab, dan kemudian pilih pengguna dalam daftar.
5. Pada halaman detail pengguna, pilih Tindakan, Setel ulang kata sandi.
6. Dalam Setel ulang kata sandi dialog, tinjau informasi dan ketika siap, pilih Setel ulang.

Tindakan ini segera menghasilkan kode konfirmasi yang dikirim ke pengguna dan menonaktifkan kata sandi pengguna saat ini dengan mengubah status pengguna menjadi `RESET_REQUIRED`. Setelah ulang kata sandi kode berlaku selama 1 jam.

## Mencari atribut pengguna

Jika Anda telah membuat kolam pengguna, Anda dapat mencari dari panel Pengguna di AWS Management Console. Anda juga dapat menggunakan Amazon Cognito [ListUsers API](#), yang menerima `Filter` parameter.

Anda dapat mencari salah satu dari atribut standar berikut. Atribut kustom tidak dapat dicari.

- nama pengguna (peka huruf besar/kecil)
- email
- phone\_number
- nama
- given\_name
- family\_name
- preferred\_username
- cognito:user\_status (dipanggil Status di konsol tersebut) (peka huruf besar/kecil)
- status (disebut Diaktifkan di konsol tersebut) (peka huruf besar/kecil)
- sub

### Note

Anda juga dapat membuat daftar pengguna dengan filter sisi klien. Filter sisi server cocok tidak lebih dari 1 atribut. Untuk pencarian lanjutan, gunakan filter sisi klien dengan `--query` parameter dari `list-users` aksi di AWS Command Line Interface. Saat Anda menggunakan filter sisi klien, `ListUsers` mengembalikan daftar paginasi dari nol atau lebih pengguna. Anda dapat menerima beberapa halaman berturut-turut dengan hasil nol. Ulangi kueri dengan setiap token pagination yang dikembalikan hingga Anda menerima nilai token pagination null, lalu tinjau hasil gabungannya.

Untuk informasi selengkapnya tentang pemfilteran sisi server dan sisi klien, lihat [Penyaringan AWS CLI keluaran](#) di AWS Command Line Interface Panduan Pengguna.

## Mencari pengguna dengan AWS Management Console

Jika Anda telah membuat kolam pengguna, Anda dapat mencari dari panel Pengguna di AWS Management Console.

Pencarian AWS Management Console selalu merupakan awalan pencarian ("dimulai dengan").

Untuk mencari pengguna di konsol Amazon Cognito

1. Masuk ke [Konsol Amazon Cognito](#). Anda mungkin diminta untuk kredensial AWS Anda.
2. Pilih Kolam Pengguna.
3. Pilih kumpulan pengguna yang ada dari daftar.
4. Pilih Pengguna tab, lalu masukkan nama pengguna pengguna di bidang pencarian. Perhatikan bahwa beberapa nilai atribut peka huruf besar/kecil (misalnya, Nama Pengguna).

Anda juga dapat menemukan pengguna dengan menyesuaikan filter pencarian untuk mempersempit cakupan ke properti pengguna lainnya, seperti Email, Nomor telepon, atau Nama belakang.

## Mencari pengguna dengan `ListUsers` API

Untuk mencari pengguna dari aplikasi Anda, gunakan Amazon Cognito [ListUsers API](#). API ini menggunakan parameter berikut:

- `AttributesToGet`: Sebuah array dari string, di mana setiap string adalah nama dari atribut pengguna yang akan dikembalikan untuk setiap pengguna dalam hasil pencarian. Untuk mengambil semua atribut, jangan sertakan `AttributesToGet` parameter atau permintaan `AttributesToGet` dengan nilai string literal `null`.
- `Filter`: Sebuah string filter dari bentuk "AttributeName Filter-Type AttributeValue". Tanda kutip dalam string filter harus diloloskan menggunakan karakter garis miring terbalik (`\`). Misalnya, "family\_name = \"Reddy\"". Jika string filter kosong, `ListUsers` mengembalikan semua pengguna di kolam pengguna.
- `AttributeName`: Nama atribut yang akan dicari. Anda hanya dapat mencari satu atribut dalam satu waktu.

**Note**

Anda hanya dapat mencari atribut standar. Atribut kustom tidak dapat dicari. Ini karena hanya atribut terindeks yang dapat dicari, dan atribut khusus tidak dapat diindeks.

- **Filter-Type**: Untuk pencocokan tepat, gunakan `=`, misalnya, `given_name = "Jon"`. Untuk kecocokan awalan ("dimulai dengan"), gunakan `^=`, misalnya, `given_name ^= "Jon"`.
- **AttributeValue**: Nilai atribut yang harus dicocokkan untuk setiap pengguna.
- **Limit**: Jumlah maksimum pengguna yang akan dikembalikan.
- **PaginationToken**: Token untuk mendapatkan lebih banyak hasil dari pencarian sebelumnya. Amazon Cognito kedaluwarsa token pagination setelah satu jam.
- **UserPoolId**: ID kolam pengguna untuk kolam pengguna tempat pencarian harus dilakukan.

Semua pencarian tidak peka huruf besar/kecil. Hasil pencarian diurutkan berdasarkan atribut yang diberi nama oleh string `AttributeName`, dalam urutan naik.

## Contoh menggunakan `ListUsersAPI`

Contoh berikut mengembalikan semua pengguna dan mencakup semua atribut.

```
{
  "AttributesToGet": null,
  "Filter": "",
  "Limit": 10,
  "UserPoolId": "us-east-1_samplepool"
}
```

Contoh berikut menampilkan semua pengguna yang nomor teleponnya dimulai dengan "+1312" dan mencakup semua atribut.

```
{
  "AttributesToGet": null,
  "Filter": "phone_number ^= \"+1312\"",
  "Limit": 10,
}
```

```
"UserPoolId": "us-east-1_samplepool"
}
```

Contoh berikut mengembalikan 10 pengguna pertama yang nama keluarganya "Reddy". Untuk setiap pengguna, hasil pencarian mencakup nama pengguna, nomor telepon, dan alamat email. Jika ada lebih dari 10 pengguna yang cocok di kolam pengguna, responsnya mencakup token pemberian nomor halaman.

```
{
  "AttributesToGet": [
    "given_name",
    "phone_number",
    "email"
  ],
  "Filter": "family_name = \"Reddy\"",
  "Limit": 10,
  "UserPoolId": "us-east-1_samplepool"
}
```

Jika contoh sebelumnya mengembalikan token pemberian nomor halaman, contoh berikut mengembalikan 10 pengguna berikutnya yang cocok dengan string filter yang sama.

```
{
  "AttributesToGet": [
    "given_name",
    "phone_number",
    "email"
  ],
  "Filter": "family_name = \"Reddy\"",
  "Limit": 10,
  "PaginationToken": "pagination_token_from_previous_search",
  "UserPoolId": "us-east-1_samplepool"
}
```

## Memulihkan akun pengguna

Parameter `AccountRecoverySetting` memungkinkan Anda untuk menyesuaikan metode mana yang dapat digunakan pengguna untuk memulihkan kata sandi mereka saat mereka memanggil API [ForgotPassword](#). `ForgotPassword` mengirimkan kode pemulihan ke email

terverifikasi atau nomor telepon terverifikasi. Kode pemulihan valid untuk satu jam. Bila Anda menentukan [AccountRecoverySetting](#) untuk kumpulan pengguna Anda, Amazon Cognito memilih tujuan pengiriman kode berdasarkan prioritas yang Anda tetapkan.

Saat Anda menentukan `AccountRecoverySetting` dan pengguna memiliki SMS MFA yang terkonfigurasi, SMS tidak dapat digunakan sebagai mekanisme pemulihan akun. Prioritas untuk pengaturan ini ditentukan dengan 1 menjadi prioritas tertinggi. Cognito mengirim verifikasi hanya ke salah satu metode yang ditentukan.

Sebagai contoh, `admin_only` adalah nilai yang digunakan ketika administrator tidak ingin pengguna memulihkan akun mereka sendiri, dan sebaliknya akan meminta mereka untuk menghubungi administrator untuk mengatur ulang akun mereka. Anda tidak dapat menggunakan `admin_only` dengan mekanisme pemulihan akun lainnya.

Jika Anda tidak menentukan `AccountRecoverySetting`, Amazon Cognito menggunakan mekanisme warisan untuk menentukan metode pemulihan kata sandi. Dalam hal ini, Cognito menggunakan telepon terverifikasi terlebih dahulu. Jika telepon terverifikasi tidak ditemukan untuk pengguna, Cognito akan kembali dan akan menggunakan email terverifikasi berikutnya.

Untuk informasi selengkapnya tentang `AccountRecoverySetting`, lihat [CreateUserPool](#) dan [UpdateUserPool](#) di Referensi API Penyedia Identitas Amazon Cognito.

## Perilaku lupa kata sandi

Dalam satu jam yang diberikan, kami mengizinkan antara 5 dan 20 percobaan untuk pengguna untuk meminta atau memasukkan kode pengaturan ulang kata sandi sebagai bagian dari lupa-sandi dan `confirm-forgot-password` tindakan. Nilai pasti tergantung pada parameter risiko yang terkait dengan permintaan. Harap dicatat bahwa perilaku ini dapat berubah.

## Mengimpor pengguna ke kolam pengguna

Ada dua cara Anda dapat mengimpor atau melakukan migrasi pengguna dari direktori pengguna atau basis data pengguna yang ada ke dalam kolam pengguna Amazon Cognito. Anda dapat melakukan migrasi pengguna saat mereka masuk menggunakan Amazon Cognito untuk pertama kalinya dengan pemicu migrasi pengguna Lambda. Dengan pendekatan ini, pengguna dapat terus menggunakan sandi yang ada dan tidak perlu mengatur ulang kata sandi setelah migrasi ke kolam pengguna Anda. Atau, Anda dapat melakukan migrasi pengguna secara massal dengan mengunggah file CSV yang berisi atribut profil pengguna untuk semua pengguna. Bagian berikut menjelaskan kedua pendekatan ini.

## Topik

- [Mengimpor pengguna ke kumpulan pengguna dengan migrasi pengguna pemicu Lambda](#)
- [Mengimpor pengguna ke kumpulan pengguna dari file CSV](#)

## Mengimpor pengguna ke kumpulan pengguna dengan migrasi pengguna pemicu Lambda

Dengan pendekatan ini, Anda dapat memigrasi pengguna dari direktori pengguna yang ada ke pangkalan pengguna secara mulus saat pengguna login untuk pertama kalinya dengan aplikasi Anda atau meminta pengaturan ulang kata sandi. Menambahkan [Memigrasi pengguna pemicu Lambda](#) berfungsi ke pangkalan pengguna Anda dan menerima metadata tentang pengguna yang mencoba masuk, dan mengembalikan informasi profil pengguna dari sumber identitas eksternal. Untuk detail dan contoh kode pemicu Lambda ini, termasuk parameter permintaan dan respons, lihat [Memigrasi parameter pemicu Lambda](#).

Sebelum Anda mulai memigrasi pengguna, buat fungsi Lambda migrasi pengguna di Akun AWS, dan atur fungsi Lambda sebagai pemicu migrasi pengguna di pangkalan pengguna Anda. Tambahkan kebijakan otorisasi ke fungsi Lambda Anda yang hanya mengizinkan prinsipal akun layanan Amazon Cognito, `cognito-idp.amazonaws.com` untuk memanggil fungsi Lambda, dan hanya dalam konteks pool pengguna Anda sendiri. Untuk informasi lebih lanjut, lihat [Menggunakan kebijakan berbasis sumber daya untuk AWS Lambda \(Kebijakan fungsi Lambda\)](#).

### Proses masuk


1. Pengguna membuka aplikasi Anda dan masuk dengan API pool pengguna Amazon Cognito atau melalui UI yang dihosting Amazon Cognito. Untuk informasi selengkapnya tentang cara memfasilitasi login dengan Amazon Cognito API, lihat [Mengintegrasikan otentikasi dan otorisasi Amazon Cognito dengan aplikasi web dan seluler](#).
2. Aplikasi Anda mengirimkan nama pengguna dan kata sandi ke Amazon Cognito. Jika aplikasi Anda memiliki UI login kustom yang dibuat dengan AWS SDK, aplikasi Anda harus menggunakan [InitiateAuth](#) atau [AdminInitiateAuth](#) dengan `USER_PASSWORD_AUTH` atau `ADMIN_USER_PASSWORD_AUTH`. Saat aplikasi Anda menggunakan salah satu alur ini, SDK akan mengirimkan sandi ke server.

#### Note

Sebelum menambahkan pemicu migrasi pengguna, aktifkan `USER_PASSWORD_AUTH` atau `ADMIN_USER_PASSWORD_AUTH` mengalir

di setelan klien aplikasi Anda. Anda harus menggunakan alur ini bukan default `USER_SRP_AUTH` mengalir. Amazon Cognito harus mengirim kata sandi ke fungsi Lambda Anda sehingga dapat memverifikasi autentikasi pengguna Anda di direktori lain. SRP mengaburkan kata sandi pengguna Anda dari fungsi Lambda Anda.

3. Amazon Cognito memeriksa apakah nama pengguna yang dikirimkan cocok dengan nama pengguna atau alias di pangkalan pengguna. Anda dapat mengatur alamat email, nomor telepon, atau nama pengguna pilihan pengguna sebagai alias di pangkalan pengguna Anda. Jika pengguna tidak ada, Amazon Cognito mengirimkan parameter, termasuk nama pengguna dan kata sandi, ke [Memigrasi pengguna pemicu Lambda](#) fungsi.
4. Anda [Memigrasi pengguna pemicu Lambda](#) fungsi memeriksa atau mengotentikasi pengguna dengan direktori pengguna atau database pengguna yang ada. Fungsi mengembalikan atribut pengguna yang disimpan Amazon Cognito di profil pengguna di pangkalan pengguna. Anda dapat mengembalikan `username` parameter hanya jika nama pengguna yang dikirimkan cocok dengan atribut alias. Jika Anda ingin pengguna terus menggunakan kata sandi yang ada, fungsi Anda akan menetapkan atribut `finalUserStatus` kepada `CONFIRMED` dalam respon Lambda. Aplikasi Anda harus mengembalikan semua "response" parameter ditampilkan di [Memigrasi parameter pemicu Lambda](#).

 Important

Jangan mencatat seluruh objek peristiwa permintaan dalam kode Lambda migrasi pengguna Anda. Permintaan objek acara ini termasuk password pengguna. Jika Anda tidak membersihkan log, kata sandi akan muncul `CloudWatchLog`.

5. Amazon Cognito membuat profil pengguna di kolam pengguna Anda, dan mengembalikan token ke klien aplikasi Anda.
6. Aplikasi Anda melakukan asupan token, menerima autentikasi pengguna, dan melanjutkan ke konten yang diminta.

Setelah memigrasi pengguna, gunakan `USER_SRP_AUTH` untuk masuk. Protokol Secure Remote Password (SRP) tidak mengirim kata sandi ke seluruh jaringan, dan memberikan manfaat keamanan atas `USER_PASSWORD_AUTH` aliran yang Anda gunakan selama migrasi.

Jika terjadi kesalahan selama migrasi, termasuk perangkat klien atau masalah jaringan, aplikasi Anda menerima respons error dari API pool pengguna Amazon Cognito. Ketika ini terjadi, Amazon Cognito mungkin atau mungkin tidak membuat akun pengguna di pangkalan pengguna Anda. Pengguna



kemudian harus mencoba masuk lagi. Jika login gagal berulang kali, coba atur ulang sandi pengguna dengan alur lupa-sandi di aplikasi Anda.

Alur lupa-kata sandi juga memanggil Anda [Memigrasi pengguna pemicu Lambda](#) fungsi dengan `UserMigration_ForgotPassword` sumber acara. Karena pengguna tidak mengirimkan kata sandi saat meminta pengaturan ulang kata sandi, Amazon Cognito tidak menyertakan kata sandi jika dikirimkan ke fungsi Lambda Anda. Fungsi Anda hanya dapat mencari pengguna di direktori pengguna yang ada dan mengembalikan atribut untuk ditambahkan ke profil pengguna di pangkalan pengguna Anda. Setelah fungsi Anda menyelesaikan pemanggilannya dan mengembalikan responsnya ke Amazon Cognito, kumpulan pengguna akan mengirimkan kode pengaturan ulang kata sandi melalui email atau SMS. Di aplikasi Anda, minta pengguna Anda untuk kode konfirmasi dan kata sandi baru, lalu kirimkan informasi itu ke Amazon Cognito dalam [ConfirmForgotPassword](#) Permintaan API. Anda juga dapat menggunakan halaman bawaan untuk alur lupa kata sandi di UI yang dihosting Amazon Cognito.

## Mengimpor pengguna ke kumpulan pengguna dari file CSV

Anda dapat mengimpor pengguna ke kolam pengguna Amazon Cognito. Informasi pengguna diimpor dari file `.csv` yang diformat khusus. Proses impor menetapkan nilai untuk semua atribut pengguna kecuali kata sandi. Impor kata sandi tidak didukung, karena praktik terbaik keamanan mengharuskan kata sandi tidak tersedia sebagai teks biasa, dan kami tidak mendukung pengimporan hash. Ini berarti pengguna Anda harus mengubah kata sandi mereka saat pertama kali mereka masuk. Jadi, pengguna Anda akan berada dalam status `RESET_REQUIRED` saat diimpor menggunakan metode ini.

Anda dapat mengatur kata sandi pengguna Anda dengan permintaan [AdminSetUserPassword](#) API yang menetapkan `Permanent` parameter ke `true`.

### Note

Tanggal pembuatan untuk setiap pengguna adalah waktu ketika pengguna tersebut diimpor ke kolam pengguna. Tanggal pembuatan bukan salah satu atribut yang diimpor.

Langkah dasarnya adalah:

1. Buat peran Amazon CloudWatch Logs di konsol AWS Identity and Access Management (IAM).
2. Buat file `.csv` impor pengguna.
3. Buat dan jalankan tugas impor pengguna.

4. Unggah file .csv impor pengguna.
5. Mulai dan jalankan tugas impor pengguna.
6. Gunakan CloudWatch untuk memeriksa log peristiwa.
7. Haruskan pengguna yang diimpor untuk mengatur ulang kata sandi mereka.

## Topik

- [Membuat peran IAM CloudWatch Log](#)
- [Membuat file CSV impor pengguna](#)
- [Membuat dan menjalankan tugas impor kolam pengguna Amazon Cognito](#)
- [Melihat hasil impor kumpulan pengguna di CloudWatch console](#)
- [Mengharuskan pengguna yang diimpor untuk mengatur ulang kata sandi mereka](#)

## Membuat peran IAM CloudWatch Log

Jika Anda menggunakan Amazon Cognito CLI atau API, Anda perlu membuat CloudWatch IAM role IAM. Prosedur berikut menjelaskan cara membuat peran IAM yang dapat digunakan Amazon Cognito untuk menulis hasil pekerjaan impor Anda ke CloudWatch Log.

### Note

Saat membuat pekerjaan impor di konsol Amazon Cognito, Anda dapat membuat peran IAM secara bersamaan. Saat Anda memilih untuk Membuat peran IAM baru, Amazon Cognito secara otomatis menerapkan kebijakan kepercayaan dan kebijakan IAM yang sesuai untuk peran tersebut.

Untuk membuat CloudWatch IAM role untuk impor kolam pengguna (AWS CLI, API)

1. Masuk ke AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Buat peran IAM baru untuk sebuah Layanan AWS. Untuk petunjuk selengkapnya, lihat [Membuat peran Layanan AWS](#) dalam Panduan AWS Identity and Access Management Pengguna.
  - a. Bila Anda memilih Kasus Penggunaan untuk jenis entitas Tepercaya, pilih layanan apa pun. Amazon Cognito saat ini tidak terdaftar dalam kasus penggunaan layanan.

- b. Dalam Tambahkan izin layar, pilih Buat kebijakan dan masukkan pernyataan kebijakan berikut. Ganti **REGION** dengan kumpulan pengguna Anda, misalnya `us-east-1`. Wilayah AWS Ganti **AKUN** dengan Akun AWS ID Anda, misalnya `111122223333`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogGroup",
        "logs:CreateLogStream",
        "logs:DescribeLogStreams",
        "logs:PutLogEvents"
      ],
      "Resource": [
        "arn:aws:logs:REGION:ACCOUNT:log-group:/aws/cognito/*"
      ]
    }
  ]
}
```

3. Karena Anda tidak memilih Amazon Cognito sebagai entitas tepercaya saat membuat peran, Anda sekarang harus mengedit hubungan kepercayaan peran secara manual. Pilih Peran dari panel navigasi konsol IAM, lalu pilih IAM role baru yang Anda buat.
4. Pilih tab Hubungan kepercayaan.
5. Pilih Edit kebijakan kepercayaan.
6. Tempelkan pernyataan kebijakan berikut ke kebijakan Edit kepercayaan, ganti teks yang ada:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "cognito-idp.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

```
}
```

7. Pilih Buat Kebijakan.
8. Perhatikan ARN peran. Anda akan memberikan ARN saat membuat pekerjaan impor.

### Membuat file CSV impor pengguna

Sebelum Anda dapat mengimpor pengguna yang ada ke kolam pengguna Anda, Anda harus membuat file nilai yang dipisahkan koma yang berisi pengguna yang ingin Anda impor, dan atribut mereka. Dari kumpulan pengguna, Anda dapat mengambil file impor pengguna dengan header yang mencerminkan skema atribut pool pengguna Anda. Anda kemudian dapat memasukkan informasi pengguna yang cocok dengan persyaratan pemformatan di [Memformat file CSV](#).

### Mengunduh header file CSV (konsol)

Gunakan prosedur berikut untuk mengunduh file header CSV.

#### Untuk mengunduh header file CSV

1. Masuk ke [Konsol Amazon Cognito](#). Anda mungkin diminta untuk kredensial AWS Anda.
2. Pilih User Pools.
3. Pilih kolam pengguna yang ada dari daftar.
4. Pilih tab Pengguna.
5. Di bagian Impor pengguna, pilih Buat pekerjaan impor.
6. Di bawah Unggah CSV, pilih tautan `template.csv` dan unduh file CSV.

### Mengunduh header file CSV (AWS CLI)

Untuk mendapatkan daftar header yang benar, jalankan perintah CLI berikut, di mana `USER_POOL_ID` adalah pengidentifikasi kolam pengguna untuk kolam pengguna tempat Anda akan mengimpor pengguna:

```
aws cognito-idp get-csv-header --user-pool-id "USER_POOL_ID"
```

#### Contoh respons:

```
{  
  "CSVHeader": [  

```

```
    "name",
    "given_name",
    "family_name",
    "middle_name",
    "nickname",
    "preferred_username",
    "profile",
    "picture",
    "website",
    "email",
    "email_verified",
    "gender",
    "birthdate",
    "zoneinfo",
    "locale",
    "phone_number",
    "phone_number_verified",
    "address",
    "updated_at",
    "cognito:mfa_enabled",
    "cognito:username"
  ],
  "UserPoolId": "USER_POOL_ID"
}
```

## Memformat file CSV

File header impor pengguna CSV yang diunduh terlihat seperti string berikut. Ini juga menyertakan atribut kustom yang Anda tambahkan ke kolam pengguna Anda.

```
cognito:username,name,given_name,family_name,middle_name,nickname,preferred_username,profile,pi
```

Edit file CSV agar menyertakan header ini dan nilai atribut untuk pengguna Anda, dan terformat menurut aturan berikut:

### Note

Untuk informasi selengkapnya tentang nilai atribut, seperti format yang tepat untuk nomor telepon, lihat [Atribut kumpulan pengguna](#).

- Baris pertama dalam file adalah baris header yang diunduh, yang berisi nama atribut pengguna.

- Urutan kolom dalam file CSV tidak masalah.
- Setiap baris setelah baris pertama berisi nilai atribut untuk pengguna.
- Semua kolom di header harus ada, tetapi Anda tidak perlu memberikan nilai di setiap kolom.
- Atribut berikut diperlukan:
  - `cognito:nama pengguna`
  - `cognito:mfa_enabled`
  - `email_verified` atau `phone_number_verified`
    - Setidaknya satu dari atribut yang diverifikasi otomatis harus `true` untuk setiap pengguna. Atribut terverifikasi otomatis adalah alamat email atau nomor telepon yang secara otomatis mengirimkan kode Amazon Cognito saat pengguna baru bergabung dengan kumpulan pengguna Anda.
    - Kolam pengguna harus memiliki setidaknya satu atribut terverifikasi otomatis, baik `email_verified` atau `phone_number_verified`. Jika kolom pengguna tidak memiliki atribut yang diverifikasi otomatis, pekerjaan impor tidak akan dimulai.
    - Jika kolom pengguna hanya memiliki satu atribut yang diverifikasi otomatis, atribut tersebut harus diverifikasi untuk setiap pengguna. Sebagai contoh, jika kolom pengguna hanya memiliki `phone_number` sebagai atribut yang diverifikasi otomatis, nilai `phone_number_verified` harus `true` untuk setiap pengguna.

#### Note

Agar pengguna dapat mengatur ulang kata sandi, mereka harus memiliki email atau nomor telepon yang diverifikasi. Amazon Cognito mengirim pesan yang berisi kode atur ulang kata sandi ke email atau nomor telepon yang ditentukan dalam file CSV. Jika pesan dikirim ke nomor telepon, pesan dikirim melalui pesan SMS. Untuk informasi selengkapnya, lihat [Memverifikasi informasi kontak saat mendaftar](#).

- Email (jika `email_verified` adalah `true`)
- `phone_number` (jika `phone_number_verified` adalah `true`)
- Atribut apa pun yang Anda tandai sebagai dibutuhkan saat Anda membuat kolom pengguna
- Nilai atribut yang berupa string tidak boleh berada dalam tanda kutip.
- Jika nilai atribut berisi koma, Anda harus meletakkan garis miring terbalik (`\`) sebelum koma. Ini karena bidang dalam file CSV dipisahkan dengan koma.
- Isi file CSV harus dalam format UTF-8 tanpa tanda urutan byte.

- Kolom `cognito:username` wajib diisi dan harus unik dalam kolam pengguna Anda. Itu bisa berupa string Unicode apa saja. Namun, itu tidak boleh berisi spasi atau tab.
- Nilai tanggal lahir, jika ada, harus dalam format `mm/dd/yyyy`. Ini berarti, misalnya, tanggal lahir 1 Februari 1985 harus dikodekan sebagai `02/01/1985`.
- Bidang `cognito:mfa_enabled` diperlukan. Jika Anda telah mengatur autentikasi multifaktor (MFA) untuk diharuskan di kolam pengguna Anda, bidang ini harus `true` untuk semua pengguna. Jika Anda telah mengatur MFA ke nonaktif, bidang ini harus `false` untuk semua pengguna. Jika Anda telah mengatur MFA menjadi opsional, bidang ini dapat berupa `true` atau `false`, tetapi tidak boleh kosong.
- Panjang baris maksimum adalah 16.000 karakter.
- Ukuran file CSV maksimumnya adalah 100 MB.
- Jumlah maksimum baris (pengguna) dalam file adalah 500.000. Maksimum ini tidak menyertakan baris header.
- Nilai bidang `updated_at` diharapkan menjadi jangka waktu dalam detik, misalnya: `1471453471`.
- Setiap spasi kosong di depan atau di belakang dalam nilai atribut akan dipangkas.

Daftar berikut adalah contoh file impor CSV untuk kumpulan pengguna tanpa atribut khusus. Skema pangkalan pengguna Anda mungkin berbeda dari contoh ini. Dalam hal ini, Anda harus memberikan nilai pengujian dalam template CSV yang Anda unduh dari pangkalan pengguna.

```
cognito:username,name,given_name,family_name,middle_name,nickname,preferred_username,profile,pi
John,,John,Doe,,,,,,,,johndoe@example.com,TRUE,,02/01/1985,,,+12345550100,TRUE,123 Any
Street,,FALSE
Jane,,Jane,Roe,,,,,,,,janeroe@example.com,TRUE,,01/01/1985,,,+12345550199,TRUE,100 Main
Street,,FALSE
```

## Membuat dan menjalankan tugas impor kolam pengguna Amazon Cognito

Bagian ini menjelaskan cara membuat dan menjalankan tugas impor kolam pengguna dengan menggunakan konsol Amazon Cognito dan AWS Command Line Interface (AWS CLI).

### Topik

- [Mengimpor pengguna dari file CSV \(konsol\)](#)
- [Mengimpor pengguna \(AWS CLI\)](#)

## Mengimpor pengguna dari file CSV (konsol)

Prosedur berikut menjelaskan cara mengimpor pengguna dari file CSV.

Untuk mengimpor pengguna dari file CSV (konsol)

1. Masuk ke [Konsol Amazon Cognito](#). Anda mungkin diminta untuk kredensial AWS Anda.
2. Pilih User Pools.
3. Pilih kolam pengguna yang ada dari daftar.
4. Pilih tab Pengguna.
5. Di bagian Impor pengguna, pilih Buat pekerjaan impor.
6. Pada halaman Buat pekerjaan impor, masukkan nama Job.
7. Pilih untuk Membuat peran IAM baru atau Menggunakan peran IAM yang ada.
  - a. Jika Anda memilih Buat peran IAM baru, masukkan nama untuk peran baru Anda. Amazon Cognito akan secara otomatis membuat peran dengan izin dan hubungan kepercayaan yang benar. Prinsip IAM yang membuat pekerjaan impor harus memiliki izin untuk membuat peran IAM.
  - b. Jika Anda memilih Gunakan peran IAM yang ada, pilih peran dari daftar di bawah pemilihan peran IAM. Peran ini harus memiliki kebijakan izin dan kepercayaan yang dijelaskan di [Membuat peran IAM CloudWatch Log](#).
8. Pilih Buat pekerjaan untuk mengirimkan pekerjaan Anda, tetapi mulai nanti. Pilih Buat dan mulai pekerjaan untuk mengirimkan pekerjaan Anda dan segera memulainya.
9. Jika Anda membuat pekerjaan Anda tetapi tidak memulainya, Anda dapat memulainya nanti. Di tab Pengguna di bawah Impor pengguna, pilih pekerjaan impor Anda, lalu pilih Mulai. Anda juga dapat mengirimkan permintaan [StartUserImportJob](#) API dari AWS SDK.
10. Pantau kemajuan pekerjaan impor pengguna Anda di tab Pengguna di bawah Impor pengguna. Jika pekerjaan Anda tidak berhasil, Anda dapat memilih nilai Status. Untuk detail tambahan, pilih Lihat CloudWatch log untuk detail selengkapnya dan tinjau masalah apa pun di konsol CloudWatch Log.

## Mengimpor pengguna (AWS CLI)

Perintah CLI berikut tersedia untuk mengimpor pengguna ke kolam pengguna:

- `create-user-import-job`



- `get-csv-header`
- `describe-user-import-job`
- `list-user-import-jobs`
- `start-user-import-job`
- `stop-user-import-job`

Untuk mendapatkan daftar opsi baris perintah untuk perintah ini, gunakan opsi baris perintah `help`. Sebagai contoh:

```
aws cognito-idp get-csv-header help
```

### Membuat pekerjaan impor pengguna

Setelah Anda membuat file CSV, buat tugas impor pengguna dengan menjalankan perintah CLI berikut, di mana *JOB\_NAME* adalah nama yang Anda pilih untuk pekerjaan tersebut, *USER\_POOL\_ID* adalah ID kolam pengguna untuk kolam pengguna tempat pengguna baru akan ditambahkan, dan *ROLE\_ARN* adalah peran ARN Anda diterima di [Membuat peran IAM CloudWatch Log](#):

```
aws cognito-idp create-user-import-job --job-name "JOB_NAME" --user-pool-id "USER_POOL_ID" --cloud-watch-logs-role-arn "ROLE_ARN"
```

*PRE\_SIGNED\_URL* yang dikembalikan dalam respons berlaku selama 15 menit. Setelah waktu itu, itu akan kedaluwarsa dan Anda harus membuat pekerjaan impor pengguna baru untuk mendapatkan URL baru.

Example Contoh respons:

```
{
  "UserImportJob": {
    "Status": "Created",
    "SkippedUsers": 0,
    "UserPoolId": "USER_POOL_ID",
    "ImportedUsers": 0,
    "JobName": "JOB_NAME",
    "JobId": "JOB_ID",
    "PreSignedUrl": "PRE_SIGNED_URL",
```

```
    "CloudWatchLogsRoleArn": "ROLE_ARN",  
    "FailedUsers": 0,  
    "CreationDate": 1470957431.965  
  }  
}
```

## Nilai status untuk pekerjaan impor pengguna

Dalam respons terhadap perintah impor pengguna, Anda akan melihat salah satu nilai Status berikut:

- **Created**- Pekerjaan telah dibuat tetapi belum dimulai.
- **Pending**- Sebuah status transisi. Anda telah memulai pekerjaan, tetapi belum mulai mengimpor pengguna.
- **InProgress**- Pekerjaan telah dimulai, dan pengguna sedang diimpor.
- **Stopping**- Anda telah menghentikan pekerjaan, tetapi pekerjaan belum berhenti mengimpor pengguna.
- **Stopped**- Anda telah menghentikan pekerjaan, dan pekerjaan telah berhenti mengimpor pengguna.
- **Succeeded**- Pekerjaan telah berhasil diselesaikan.
- **Failed**- Pekerjaan dihentikan karena kesalahan.
- **Expired**- Anda membuat pekerjaan, tetapi tidak memulai pekerjaan dalam waktu 24-48 jam. Semua data yang terkait dengan pekerjaan telah dihapus, dan pekerjaan tidak dapat dimulai.

## Mengunggah file CSV

Gunakan `curl` perintah berikut untuk mengunggah file CSV yang berisi data pengguna Anda ke URL yang telah ditentukan sebelumnya yang Anda peroleh dari `responscreeate-user-import-job` perintah.

```
curl -v -T "PATH_TO_CSV_FILE" -H "x-amz-server-side-encryption:aws:kms"  
"PRE_SIGNED_URL"
```

Dalam output dari perintah ini, cari frasa "We are completely uploaded and fine". Frasa ini menunjukkan bahwa file telah berhasil diunggah.

## Menjelaskan pekerjaan impor pengguna

Untuk mendapatkan deskripsi pekerjaan impor pengguna Anda, gunakan perintah berikut, di mana *USER\_POOL\_ID* adalah ID kolam pengguna Anda, dan *JOB\_ID* adalah ID pekerjaan yang dikembalikan saat Anda membuat pekerjaan impor pengguna.

```
aws cognito-idp describe-user-import-job --user-pool-id "USER_POOL_ID" --job-id "JOB_ID"
```

Example Contoh respons:

```
{
  "UserImportJob": {
    "Status": "Created",
    "SkippedUsers": 0,
    "UserPoolId": "USER_POOL_ID",
    "ImportedUsers": 0,
    "JobName": "JOB_NAME",
    "JobId": "JOB_ID",
    "PreSignedUrl": "PRE_SIGNED_URL",
    "CloudWatchLogsRoleArn": "ROLE_ARN",
    "FailedUsers": 0,
    "CreationDate": 1470957431.965
  }
}
```

Dalam contoh keluaran sebelumnya, *PRE\_SIGNED\_URL* adalah URL tempat Anda mengunggah file CSV. *ROLE\_ARN* adalah CloudWatch ARN peran yang Anda terima saat membuat peran.

## Mencantumkan pekerjaan impor pengguna Anda

Untuk membuat daftar pekerjaan impor pengguna Anda, gunakan perintah berikut:

```
aws cognito-idp list-user-import-jobs --user-pool-id "USER_POOL_ID" --max-results 2
```

Example Contoh respons:

```
{
  "UserImportJobs": [
    {
      "Status": "Created",
      "SkippedUsers": 0,

```

```

    "UserPoolId": "USER_POOL_ID",
    "ImportedUsers": 0,
    "JobName": "JOB_NAME",
    "JobId": "JOB_ID",
    "PreSignedUrl": "PRE_SIGNED_URL",
    "CloudWatchLogsRoleArn": "ROLE_ARN",
    "FailedUsers": 0,
    "CreationDate": 1470957431.965
  },
  {
    "CompletionDate": 1470954227.701,
    "StartDate": 1470954226.086,
    "Status": "Failed",
    "UserPoolId": "USER_POOL_ID",
    "ImportedUsers": 0,
    "SkippedUsers": 0,
    "JobName": "JOB_NAME",
    "CompletionMessage": "Too many users have failed or been skipped during the
import.",
    "JobId": "JOB_ID",
    "PreSignedUrl": "PRE_SIGNED_URL",
    "CloudWatchLogsRoleArn": "ROLE_ARN",
    "FailedUsers": 5,
    "CreationDate": 1470953929.313
  }
],
"PaginationToken": "PAGINATION_TOKEN"
}

```

Pekerjaan terdaftar dalam urutan kronologis dari yang terakhir dibuat hingga yang pertama dibuat. String `PAGINATION_TOKEN` setelah pekerjaan kedua menunjukkan bahwa ada hasil tambahan untuk perintah daftar ini. Untuk membuat daftar hasil tambahan, gunakan opsi `--pagination-token` sebagai berikut:

```
aws cognito-idp list-user-import-jobs --user-pool-id "USER_POOL_ID" --max-results 10 --
pagination-token "PAGINATION_TOKEN"
```

## Memulai pekerjaan impor pengguna

Untuk memulai pekerjaan impor pengguna, gunakan perintah berikut:

```
aws cognito-idp start-user-import-job --user-pool-id "USER_POOL_ID" --job-id "JOB_ID"
```

Hanya satu tugas impor yang dapat aktif pada satu waktu per akun.

Example Contoh respons:

```
{
  "UserImportJob": {
    "Status": "Pending",
    "StartDate": 1470957851.483,
    "UserPoolId": "USER_POOL_ID",
    "ImportedUsers": 0,
    "SkippedUsers": 0,
    "JobName": "JOB_NAME",
    "JobId": "JOB_ID",
    "PreSignedUrl": "PRE_SIGNED_URL",
    "CloudWatchLogsRoleArn": "ROLE_ARN",
    "FailedUsers": 0,
    "CreationDate": 1470957431.965
  }
}
```

Menghentikan pekerjaan impor pengguna

Untuk menghentikan pekerjaan impor pengguna saat sedang berlangsung, gunakan perintah berikut. Setelah Anda menghentikan pekerjaan, itu tidak dapat dimulai ulang.

```
aws cognito-idp stop-user-import-job --user-pool-id "USER_POOL_ID" --job-id "JOB_ID"
```

Example Contoh respons:

```
{
  "UserImportJob": {
    "CompletionDate": 1470958050.571,
    "StartDate": 1470958047.797,
    "Status": "Stopped",
    "UserPoolId": "USER_POOL_ID",
    "ImportedUsers": 0,
    "SkippedUsers": 0,
    "JobName": "JOB_NAME",
    "CompletionMessage": "The Import Job was stopped by the developer.",
    "JobId": "JOB_ID",
    "PreSignedUrl": "PRE_SIGNED_URL",
    "CloudWatchLogsRoleArn": "ROLE_ARN",
  }
}
```

```
    "FailedUsers": 0,  
    "CreationDate": 1470957972.387  
  }  
}
```

Melihat hasil impor kumpulan pengguna di CloudWatch console

Anda dapat melihat hasil pekerjaan impor Anda di konsol Amazon CloudWatch.

Topik

- [Melihat hasil](#)
- [Menafsirkan hasil](#)

Melihat hasil

Langkah-langkah berikut menjelaskan cara melihat hasil impor kolam pengguna.

Untuk melihat hasil impor kolam pengguna

1. Masuk ke AWS Management Console dan buka konsol CloudWatch di <https://console.aws.amazon.com/cloudwatch/>.
2. Pilih Log.
3. Pilih grup log untuk pekerjaan impor kolam pengguna Anda. Nama grup log berada di formulir / aws/cognito/userpools/*USER\_POOL\_ID*/*USER\_POOL\_NAME*.
4. Pilih log untuk pekerjaan impor pengguna yang baru saja Anda jalankan. Nama log dalam bentuk *JOB\_ID*/*JOB\_NAME*. Hasil di log merujuk ke pengguna Anda berdasarkan nomor baris. Tidak ada data pengguna yang ditulis ke log. Untuk setiap pengguna, baris yang mirip dengan berikut akan muncul:
  - [SUCCEEDED] Line Number 5956 - The import succeeded.
  - [SKIPPED] Line Number 5956 - The user already exists.
  - [FAILED] Line Number 5956 - The User Record does not set any of the auto verified attributes to true. (Example: email\_verified to true).

Menafsirkan hasil

Pengguna yang berhasil diimpor memiliki status yang diatur ke "PasswordReset".

Dalam kasus berikut, pengguna tidak akan diimpor, tetapi pekerjaan impor akan dilanjutkan:

- Tidak ada atribut terverifikasi otomatis yang disetel ke `true`.
- Data pengguna tidak cocok dengan skema.
- Pengguna tidak dapat diimpor karena kesalahan internal.

Dalam kasus berikut, pekerjaan impor akan gagal:


- Peran Amazon CloudWatch Logs tidak dapat diasumsikan, tidak memiliki kebijakan akses yang benar, atau telah dihapus.
- Kolam pengguna telah dihapus.
- Amazon Cognito tidak dapat menguraikan file `.csv`.

Mengharuskan pengguna yang diimpor untuk mengatur ulang kata sandi mereka

Pertama kali setiap pengguna yang diimpor masuk dan memasukkan kata sandi apa pun, mereka diharuskan memasukkan kata sandi baru. Prosedur berikut menjelaskan pengalaman pengguna dalam aplikasi khusus dengan pengguna lokal setelah Anda mengimpor file CSV. Jika pengguna Anda masuk dengan UI yang dihosting, Amazon Cognito akan meminta mereka untuk menyetel kata sandi baru saat pertama kali masuk.

Mengharuskan pengguna yang diimpor untuk mengatur ulang kata sandi mereka

1. Di aplikasi Anda, coba login secara diam-diam untuk pengguna saat ini dengan `InitiateAuth` menggunakan kata sandi acak.
2. Amazon Cognito mengembalikan `NotAuthorizedException` kapan `PreventUserExistenceErrors` diaktifkan. Jika tidak, ia mengembalikan `PasswordResetRequiredException`.
3. Aplikasi Anda membuat permintaan `ForgotPassword` API dan mengatur ulang kata sandi pengguna.
  - a. Aplikasi mengirimkan nama pengguna dalam permintaan `ForgotPassword` API.
  - b. Amazon Cognito mengirimkan kode ke email atau nomor telepon yang diverifikasi. Tujuan tergantung pada nilai yang Anda berikan untuk `email_verified` dan `phone_number_verified` dalam file CSV Anda. Respons terhadap `ForgotPassword` permintaan menunjukkan tujuan kode.

 Note


Kumpulan pengguna Anda harus dikonfigurasi untuk memverifikasi email atau nomor telepon. Untuk informasi selengkapnya, lihat [Mendaftar dan mengonfirmasi akun pengguna](#).

- c. Aplikasi Anda menampilkan pesan kepada pengguna Anda untuk memeriksa lokasi pengiriman kode, dan meminta pengguna untuk memasukkan kode dan kata sandi baru.
- d. Pengguna memasukkan kode dan kata sandi baru di dalam aplikasi.
- e. Aplikasi mengirimkan kode dan kata sandi baru dalam permintaan `ConfirmForgotPassword` API.
- f. Aplikasi Anda mengarahkan pengguna Anda untuk masuk.

## Atribut kumpulan pengguna

Atribut adalah potongan informasi yang membantu Anda mengidentifikasi pengguna individu, seperti nama, alamat email, dan nomor telepon. Sebuah kumpulan pengguna baru memiliki satu set atribut standar default. Anda juga dapat menambahkan atribut kustom ke definisi kumpulan pengguna Anda di AWS Management Console. Topik ini menjelaskan atribut tersebut secara detail dan memberi Anda tips tentang cara menyiapkan kolam pengguna Anda.

Jangan menyimpan semua informasi tentang pengguna Anda dalam atribut. Misalnya, simpan data pengguna yang sering berubah, seperti statistik penggunaan atau skor game, di penyimpanan data terpisah, seperti Amazon Cognito Sync atau Amazon DynamoDB.

 Note

Beberapa dokumentasi dan standar mengacu pada atribut sebagai anggota.

### Topik

- [Atribut standar](#)
- [Nama pengguna dan nama pengguna pilihan](#)
- [Menyesuaikan atribut masuk](#)
- [Atribut kustom](#)



- [Izin dan cakupan atribut](#)

## Atribut standar

Amazon Cognito menetapkan semua pengguna satu set atribut standar berdasarkan spesifikasi [OpenID Connect](#). Secara default, nilai atribut standar dan kustom dapat berupa string apa pun dengan panjang hingga 2048 karakter, tetapi beberapa nilai atribut memiliki batasan format.

Atribut standar adalah:

- address
- birthdate
- email
- family\_name
- gender
- given\_name
- locale
- middle\_name
- name
- nickname
- phone\_number
- picture
- preferred\_username
- profile
- sub
- updated\_at
- website
- zoneinfo

Kecuali untuk `sub`, atribut standar opsional secara default untuk semua pengguna. Untuk membuat atribut diperlukan, selama proses pembuatan kumpulan pengguna, pilih kotak centang Diperlukan di sebelah atribut. Amazon Cognito menetapkan nilai pengenalan pengguna unik untuk setiap atribut pengguna. `sub` Hanya atribut email dan `phone_number` yang dapat diverifikasi.

**Note**

Ketika Anda menandai atribut standar sebagai Diperlukan, pengguna tidak dapat mendaftar kecuali mereka memberikan nilai untuk atribut tersebut. Untuk membuat pengguna dan tidak memberikan nilai untuk atribut yang diperlukan, administrator dapat menggunakan [AdminCreateUser](#) API. Setelah membuat kumpulan pengguna, Anda tidak dapat mengganti atribut antara required dan not required.

**Detail atribut standar dan batasan format****tanggal lahir**

Nilai harus tanggal 10 karakter yang valid dalam format YYYY-MM-DD.

**Email**

Pengguna dan administrator dapat memverifikasi nilai alamat email.

Administrator dengan Akun AWS izin yang tepat dapat mengubah alamat email pengguna dan juga menandainya sebagai terverifikasi. Tandai alamat email sebagai diverifikasi dengan [AdminUpdateUserAttributes](#) API atau perintah [admin-update-user-attributes](#) AWS Command Line Interface (AWS CLI). Dengan perintah ini, administrator dapat mengubah `email_verified` atribut ke `true`. Anda juga dapat mengedit pengguna di tab Pengguna AWS Management Console untuk menandai alamat email sebagai terverifikasi.

Nilai harus berupa string alamat email yang valid mengikuti format email standar dengan simbol `@` dan domain, hingga 2048 karakter panjangnya.

**phone\_number**

Pengguna harus memberikan nomor telepon jika otentikasi multi-faktor SMS (MFA) aktif. Untuk informasi selengkapnya, lihat [Menambahkan MFA ke kumpulan pengguna](#).

Pengguna dan administrator dapat memverifikasi nilai nomor telepon.

Administrator dengan Akun AWS izin yang tepat dapat mengubah nomor telepon pengguna dan juga menandainya sebagai terverifikasi. Tandai nomor telepon sebagai diverifikasi dengan [AdminUpdateUserAttributes](#) API atau [admin-update-user-attributes](#) AWS CLI perintah. Dengan perintah ini, administrator dapat mengubah `phone_number_verified` atribut ke `true`. Anda

juga dapat mengedit pengguna di tab Pengguna AWS Management Console untuk menandai nomor telepon sebagai terverifikasi.

 Important

Nomor telepon harus mengikuti aturan format ini: Nomor telepon harus dimulai dengan tanda plus (+), segera diikuti oleh kode negara. Nomor telepon hanya boleh berisi tanda + dan angka. Hapus karakter lain dari nomor telepon, seperti tanda kurung, spasi, atau tanda hubung (-) sebelum Anda mengirimkan nilai ke layanan. Misalnya, nomor telepon yang berbasis di Amerika Serikat harus mengikuti format ini: **+14325551212**.

## preferred\_username


Anda dapat memilih `preferred_username` sesuai kebutuhan atau sebagai alias, tetapi tidak keduanya. Jika alias `preferred_username` adalah, Anda dapat membuat permintaan ke operasi [UpdateUserAttributes](#) API dan menambahkan nilai atribut setelah Anda mengonfirmasi pengguna.

## sub

Indeks dan cari pengguna Anda berdasarkan sub atribut. `subAtribut` adalah pengidentifikasi pengguna unik dalam setiap kumpulan pengguna. Pengguna dapat mengubah atribut seperti `phone_number` dan `email`. `subAtribut` memiliki nilai tetap. Untuk informasi selengkapnya tentang menemukan pengguna, lihat [Mengelola dan mencari akun pengguna](#).

## Lihat atribut yang diperlukan

Gunakan prosedur berikut untuk melihat atribut yang diperlukan untuk kumpulan pengguna tertentu.

 Note

Anda tidak dapat mengubah atribut yang diperlukan setelah membuat kumpulan pengguna.

## Untuk melihat atribut yang diperlukan

1. Pergi ke [Amazon Cognito](#) di AWS Management Console. Jika konsol meminta Anda, masukkan AWS kredensial Anda.

2. Pilih Kolam Pengguna.
3. Pilih kumpulan pengguna yang ada dari daftar.
4. Pilih tab Sign-up experience.
5. Di bagian Atribut wajib, lihat atribut yang diperlukan dari kumpulan pengguna Anda.

## Nama pengguna dan nama pengguna pilihan

Nilai `username` adalah atribut terpisah dan tidak sama dengan atribut `name`. Setiap pengguna memiliki `username` atribut. Amazon Cognito secara otomatis menghasilkan nama pengguna untuk pengguna federasi. Anda harus memberikan `username` atribut untuk membuat pengguna lokal di direktori Amazon Cognito. Setelah Anda membuat pengguna, Anda tidak dapat mengubah nilai `username` atribut.

Pengembang dapat menggunakan `preferred_username` atribut untuk memberikan nama pengguna pengguna yang dapat mereka ubah. Untuk informasi selengkapnya, lihat [Menyesuaikan atribut masuk](#).

Jika aplikasi Anda tidak memerlukan nama pengguna, Anda tidak perlu meminta pengguna untuk menyediakannya. Aplikasi Anda dapat membuat nama pengguna unik untuk pengguna di latar belakang. Ini dapat berguna jika Anda ingin pengguna mendaftar dan masuk dengan alamat email dan kata sandi. Untuk informasi selengkapnya, lihat [Menyesuaikan atribut masuk](#).

`username` harus unik dalam kolam pengguna. A `username` dapat digunakan kembali, tetapi hanya setelah Anda menghapusnya dan tidak lagi digunakan. Untuk informasi tentang batasan string ke `username` atribut, lihat properti nama pengguna permintaan API. [SignUp](#)

## Menyesuaikan atribut masuk

Saat membuat kumpulan pengguna, Anda dapat mengatur atribut nama pengguna jika ingin pengguna dapat mendaftar dan masuk dengan alamat email atau nomor telepon sebagai nama pengguna mereka. Atau, Anda dapat mengatur atribut alias untuk memberikan pilihan kepada pengguna Anda: mereka dapat menyertakan beberapa atribut saat mereka mendaftar, lalu masuk dengan nama pengguna, nama pengguna pilihan, alamat email, atau nomor telepon.

### Important

Setelah membuat kumpulan pengguna, Anda tidak dapat mengubah pengaturan ini.

## Cara memilih antara atribut alias dan atribut nama pengguna

Kebutuhan Anda	Atribut alias	Atribut nama pengguna
Pengguna memiliki beberapa atribut masuk	Ya <sup>1</sup>	Tidak <sup>2</sup>
Pengguna harus memverifikasi alamat email atau nomor telepon sebelum mereka dapat masuk dengannya	Ya	Tidak
Mendaftar pengguna dengan alamat email duplikat atau nomor telepon dan mencegah <code>UsernameExistsException</code> kesalahan <sup>3</sup>	Ya	Tidak
Dapat menetapkan nilai atribut alamat email atau nomor telepon yang sama ke lebih dari satu pengguna	Ya <sup>3/</sup>	Tidak

<sup>1</sup> Atribut masuk yang tersedia adalah nama pengguna, alamat email, nomor telepon, dan nama pengguna pilihan.

<sup>2</sup> Dapat masuk dengan alamat email atau nomor telepon.

<sup>3</sup> Kumpulan pengguna Anda tidak menghasilkan `UsernameExistsException` kesalahan saat pengguna mendaftar dengan alamat email atau nomor telepon yang berpotensi duplikat, tetapi tidak ada nama pengguna. Perilaku ini tidak tergantung pada Mencegah kesalahan keberadaan nama pengguna, yang berlaku untuk operasi masuk, tetapi tidak mendaftar.

Hanya pengguna terakhir yang telah memverifikasi atribut yang dapat masuk dengannya.

### Opsi 1: Beberapa atribut masuk (atribut alias)

Anda dapat mengaktifkan alias jika Anda ingin mengizinkan pengguna Anda memilih untuk memasukkan nama pengguna atau nilai atribut lainnya saat mereka masuk. Secara default,

pengguna masuk dengan nama pengguna dan kata sandi mereka. Nama pengguna adalah nilai tetap yang tidak dapat diubah pengguna. Jika Anda menandai atribut sebagai alias, pengguna dapat masuk dengan atribut tersebut sebagai pengganti nama pengguna. Anda dapat menandai alamat email, nomor telepon, dan atribut nama pengguna pilihan sebagai alias. Misalnya, jika Anda memilih alamat email dan nomor telepon sebagai alias untuk kumpulan pengguna, pengguna di kumpulan pengguna tersebut dapat masuk dengan nama pengguna, alamat email, atau nomor telepon mereka, bersama dengan kata sandi mereka.

Untuk memilih atribut alias, pilih Nama pengguna dan setidaknya satu opsi masuk tambahan saat Anda membuat kumpulan pengguna.

#### Note

Saat Anda mengonfigurasi kumpulan pengguna agar tidak peka huruf besar/kecil, pengguna dapat menggunakan huruf kecil atau huruf besar untuk mendaftarkan atau masuk dengan alias mereka. Untuk informasi selengkapnya, lihat [CreateUserPool](#) di Referensi API kumpulan pengguna Amazon Cognito.

Jika Anda memilih alamat email sebagai alias, Amazon Cognito tidak menerima nama pengguna yang cocok dengan format alamat email yang valid. Demikian pula, jika Anda memilih nomor telepon sebagai alias, Amazon Cognito tidak menerima nama pengguna untuk kumpulan pengguna yang cocok dengan format nomor telepon yang valid.

#### Note

Nilai alias harus unik di kolam pengguna. Jika Anda mengonfigurasi alias untuk alamat email atau nomor telepon, nilai yang Anda berikan dapat berada dalam status terverifikasi hanya dalam satu akun. Selama pendaftaran, jika pengguna Anda memberikan alamat email atau nomor telepon sebagai nilai alias dan pengguna lain telah menggunakan nilai alias itu, pendaftaran berhasil. Namun, ketika pengguna mencoba mengonfirmasi akun dengan email ini (atau nomor telepon) dan memasukkan kode yang valid, Amazon Cognito mengembalikan kesalahan `AliasExistsException`. Kesalahan menunjukkan kepada pengguna bahwa akun dengan alamat email ini (atau nomor telepon) sudah ada. Pada titik ini, pengguna dapat meninggalkan upaya mereka untuk membuat akun baru dan sebagai gantinya mencoba mengatur ulang kata sandi untuk akun lama. Jika pengguna terus membuat akun baru, aplikasi Anda harus memanggil `ConfirmSignUp` API dengan `forceAliasCreation` opsi.

`ConfirmSignUp` dengan `forceAliasCreation` memindahkan alias dari akun sebelumnya ke akun yang baru dibuat, dan menandai atribut yang tidak diverifikasi di akun sebelumnya.

Nomor telepon dan alamat email hanya menjadi alias aktif untuk pengguna setelah pengguna Anda memverifikasi nomor telepon dan alamat email. Kami menyarankan Anda memilih verifikasi otomatis alamat email dan nomor telepon jika Anda menggunakannya sebagai alias.

Pilih atribut alias untuk mencegah `UsernameExistsException` kesalahan pada atribut alamat email dan nomor telepon saat pengguna Anda mendaftar.

Aktifkan `preferred_username` atribut sehingga pengguna Anda dapat mengubah nama pengguna yang mereka gunakan untuk masuk sementara nilai `username` atributnya tidak berubah. Jika Anda ingin mengatur pengalaman pengguna ini, kirimkan `username` nilai baru sebagai `preferred_username` dan pilih `preferred_username` sebagai alias. Kemudian pengguna dapat masuk dengan nilai baru yang mereka masukkan. Jika Anda memilih `preferred_username` sebagai alias, pengguna Anda dapat memberikan nilai hanya ketika mereka mengkonfirmasi akun. Mereka tidak dapat memberikan nilai selama pendaftaran.

Saat pengguna mendaftar dengan nama pengguna, Anda dapat memilih apakah mereka dapat masuk dengan satu atau beberapa alias berikut.

- Alamat email terverifikasi
- Nomor telepon terverifikasi
- Nama pengguna yang disukai

Setelah pengguna mendaftar, mereka dapat mengubah alias ini.

#### Important

Jika kumpulan pengguna Anda mendukung login dengan alias dan Anda ingin mengotorisasi atau mencari pengguna, jangan mengidentifikasi pengguna Anda dengan atribut login mereka. Pengenal pengguna nilai tetap sub adalah satu-satunya indikator yang konsisten dari identitas pengguna Anda.

Sertakan langkah-langkah berikut saat Anda membuat kumpulan pengguna sehingga pengguna dapat masuk dengan alias.

Untuk mengonfigurasi kumpulan pengguna sehingga pengguna dapat masuk dengan nama pengguna pilihan

1. Pergi ke [Amazon Cognito](#) di AWS Management Console. Jika konsol meminta Anda, masukkan AWS kredensial Anda.
2. Pilih Kolam Pengguna.
3. Di sudut kanan atas halaman, pilih Buat kumpulan pengguna untuk memulai panduan pembuatan kumpulan pengguna.
4. Di Mengonfigurasi pengalaman masuk, pilih identitas Jenis Penyedia yang ingin Anda kaitkan dengan kumpulan pengguna Anda.
5. Di bawah opsi masuk kumpulan pengguna Cognito, pilih kombinasi apa pun dari Nama Pengguna, Email, dan Nomor Telepon.
6. Di bawah Persyaratan nama pengguna, pilih Izinkan pengguna masuk dengan nama pengguna pilihan sehingga pengguna Anda dapat menetapkan nama pengguna alternatif saat mereka masuk.
7. Pilih Berikutnya, lalu selesaikan semua langkah di wizard.

Opsi 2: Alamat email atau nomor telepon sebagai atribut masuk (atribut nama pengguna)

Ketika pengguna mendaftar dengan alamat email atau nomor telepon sebagai nama pengguna mereka, Anda dapat memilih apakah mereka dapat mendaftar hanya dengan alamat email, hanya nomor telepon, atau salah satu.

Untuk memilih atribut nama pengguna, jangan pilih Nama pengguna sebagai opsi masuk saat Anda membuat kumpulan pengguna.

Alamat email atau nomor telepon harus unik, dan itu harus belum digunakan oleh pengguna lain. Itu tidak harus diverifikasi. Setelah pengguna mendaftar dengan alamat email atau nomor telepon, pengguna tidak dapat membuat akun baru dengan alamat email atau nomor telepon yang sama. Pengguna hanya dapat menggunakan kembali akun yang ada dan mengatur ulang kata sandi akun, jika diperlukan. Namun, pengguna dapat mengubah alamat email atau nomor telepon menjadi alamat email atau nomor telepon baru. Jika alamat email atau nomor telepon belum digunakan, itu menjadi nama pengguna baru.



**Note**

Jika pengguna mendaftar dengan alamat email sebagai nama pengguna mereka, mereka dapat mengubah nama pengguna ke alamat email lain, tetapi mereka tidak dapat mengubahnya menjadi nomor telepon. Jika mereka mendaftar dengan nomor telepon, mereka dapat mengubah nama pengguna ke nomor telepon lain, tetapi mereka tidak dapat mengubahnya menjadi alamat email.

Gunakan langkah-langkah berikut selama proses pembuatan kumpulan pengguna untuk mengatur pendaftaran dan masuk dengan alamat email atau nomor telepon.

Untuk mengonfigurasi kumpulan pengguna untuk mendaftar dan masuk dengan alamat email atau nomor telepon

1. Pergi ke [Amazon Cognito](#) di AWS Management Console. Jika konsol meminta Anda, masukkan AWS kredensial Anda.
2. Pilih Kolam Pengguna.
3. Di sudut kanan atas halaman, pilih Buat kumpulan pengguna untuk memulai panduan pembuatan kumpulan pengguna.
4. Di bawah opsi masuk kumpulan pengguna Cognito, pilih kombinasi Email dan Nomor Telepon yang mewakili atribut yang dapat digunakan pengguna untuk masuk.
5. Pilih Berikutnya, dan kemudian selesaikan langkah-langkah yang tersisa di wizard.

**Note**

Anda tidak perlu menandai alamat email atau nomor telepon sebagai atribut yang diperlukan untuk kumpulan pengguna Anda.

Untuk menerapkan opsi 2 di aplikasi

1. Panggil API `CreateUserPool` untuk membuat kolam pengguna Anda. Atur parameter `UserNameAttributes` menjadi `phone_number`, `email`, atau `phone_number | email`.
2. Panggil API `SignUp` dan berikan alamat email atau nomor telepon di parameter `username` dari API. API ini melakukan hal berikut:

- Jika `username` string dalam format alamat email yang valid, kumpulan pengguna secara otomatis mengisi `email` atribut pengguna dengan `username` nilai.
- Jika string `username` dalam format nomor telepon yang valid, kolom pengguna secara otomatis mengisi atribut `phone_number` dari pengguna dengan nilai `username`.
- Jika format `username` string tidak dalam alamat email atau format nomor telepon, `SignUp` API akan mengembalikan pengecualian.
- `SignUpAPI` menghasilkan UUID persisten untuk pengguna Anda, dan menggunakannya secara internal sebagai atribut nama pengguna yang tidak dapat diubah. UUID ini memiliki nilai yang sama dengan `klaim sub` dalam token identitas pengguna.
- Jika `username` string berisi alamat email atau nomor telepon yang sudah digunakan, `SignUp` API akan mengembalikan pengecualian.

Anda dapat menggunakan alamat email atau nomor telepon sebagai alias menggantikan nama pengguna di semua API kecuali API `ListUsers`. Saat Anda menelepon `ListUsers`, Anda dapat mencari berdasarkan `phone_number` atribut `email` atau. Jika Anda mencari `username`, Anda harus memberikan nama pengguna yang sebenarnya, bukan alias.


## Atribut kustom

Anda dapat menambahkan hingga 50 atribut kustom ke kumpulan pengguna Anda. Anda dapat menentukan panjang minimum dan/atau maksimum untuk atribut kustom. Namun, panjang maksimum untuk setiap atribut kustom tidak boleh lebih dari 2048 karakter.

Setiap atribut kustom memiliki karakteristik sebagai berikut:

- Anda dapat mendefinisikannya sebagai string atau angka. Amazon Cognito menulis nilai atribut khusus ke token ID hanya sebagai string.
- Anda tidak dapat meminta pengguna memberikan nilai untuk atribut tersebut.
- Anda tidak dapat menghapus atau mengubahnya setelah Anda menambahkannya ke kumpulan pengguna.
- Panjang karakter dari nama atribut berada dalam batas yang diterima Amazon Cognito. Untuk informasi selengkapnya, lihat [Kuota di Amazon Cognito](#).
- Itu bisa berubah atau tidak berubah. Anda hanya dapat menulis nilai ke atribut yang tidak dapat diubah saat Anda membuat pengguna. Anda dapat mengubah nilai atribut yang dapat berubah jika

klien aplikasi Anda memiliki izin tulis ke atribut tersebut. Untuk informasi selengkapnya, lihat [Izin dan cakupan atribut](#).

 Note

Dalam kode Anda, dan dalam pengaturan aturan untuk [Menggunakan kontrol akses berbasis peran](#), atribut khusus memerlukan custom: awalan untuk membedakannya dari atribut standar.

Anda juga dapat menambahkan atribut pengembang saat membuat kumpulan pengguna, di `SchemaAttributes` properti [CreateUserPool](#). Atribut pengembang memiliki dev: awalan. Anda hanya dapat memodifikasi atribut pengembang pengguna dengan AWS kredensial. Atribut pengembang adalah fitur lama yang diganti Amazon Cognito dengan izin baca-tulis klien aplikasi.

Gunakan prosedur berikut untuk membuat atribut kustom baru.

Untuk menambahkan atribut kustom menggunakan konsol

1. Pergi ke [Amazon Cognito](#) di AWS Management Console. Jika konsol meminta Anda, masukkan AWS kredensial Anda.
2. Pilih Kolam Pengguna.
3. Pilih kumpulan pengguna yang ada dari daftar.
4. Pilih tab Sign-up experience, dan di bagian Atribut kustom, pilih Tambahkan atribut kustom.
5. Pada halaman Tambahkan atribut kustom, berikan detail berikut tentang atribut baru:
  - Masukkan Nama.
  - Pilih Jenis String atau Angka.
  - Masukkan panjang string Min atau nilai angka.
  - Masukkan panjang string Max atau nilai angka.
  - Pilih Mutable jika Anda ingin memberikan izin kepada pengguna untuk mengubah nilai atribut kustom setelah mereka menetapkan nilai awal.
6. Pilih Simpan perubahan.

## Izin dan cakupan atribut

Untuk setiap klien aplikasi, Anda dapat mengatur izin baca dan tulis untuk setiap atribut pengguna. Dengan cara ini, Anda dapat mengontrol akses yang dimiliki aplikasi apa pun untuk membaca dan memodifikasi setiap atribut yang Anda simpan untuk pengguna Anda. Misalnya, Anda mungkin memiliki atribut kustom yang menunjukkan apakah pengguna adalah pelanggan yang membayar atau bukan. Aplikasi Anda mungkin dapat melihat atribut ini tetapi tidak mengubahnya secara langsung. Sebaliknya, Anda akan memperbarui atribut ini menggunakan alat administratif atau proses latar belakang. Anda dapat menyetel izin untuk atribut pengguna dari konsol Amazon Cognito, Amazon Cognito API, atau AWS CLI. Secara default, atribut kustom baru tidak tersedia sampai Anda menetapkan izin baca dan tulis untuk mereka. Secara default, saat Anda membuat klien aplikasi baru, Anda memberikan izin baca dan tulis aplikasi untuk semua atribut standar dan kustom. Untuk membatasi aplikasi Anda hanya pada jumlah informasi yang diperlukan, tetapkan izin khusus ke atribut dalam konfigurasi klien aplikasi Anda.

Sebagai praktik terbaik, tentukan izin baca dan tulis atribut saat Anda membuat klien aplikasi. Berikan akses klien aplikasi Anda ke set minimum atribut pengguna yang Anda perlukan untuk pengoperasian aplikasi Anda.

### Note

[DescribeUserPoolClient](#) hanya mengembalikan nilai untuk `ReadAttributes` dan `WriteAttributes` saat Anda mengonfigurasi izin klien aplikasi selain default.

Untuk memperbarui izin atribut ( )AWS Management Console

1. Pergi ke [Amazon Cognito](#) di AWS Management Console. Jika konsol meminta Anda, masukkan AWS kredensial Anda.
2. Pilih Kolam Pengguna.
3. Pilih kumpulan pengguna yang ada dari daftar.
4. Pilih tab Integrasi aplikasi, dan di bagian Klien aplikasi, pilih klien aplikasi dari daftar.
5. Di bagian Atribut baca dan tulis izin, pilih Edit.
6. Pada halaman izin baca dan tulis atribut Edit, konfigurasi izin baca dan tulis Anda, lalu pilih Simpan perubahan.

Ulangi langkah-langkah ini untuk setiap klien aplikasi yang menggunakan atribut kustom.

Untuk setiap aplikasi, Anda dapat menandai atribut sebagai dapat dibaca atau ditulis. Ini berlaku untuk atribut standar dan kustom. Aplikasi Anda dapat mengambil nilai atribut yang Anda tandai sebagai dapat dibaca, dan dapat menetapkan atau mengubah nilai atribut yang Anda tandai sebagai dapat ditulis. Jika aplikasi Anda mencoba menetapkan nilai untuk atribut yang tidak diizinkan untuk ditulis, Amazon Cognito akan kembali. `NotAuthorizedException` [GetUser](#) permintaan menyertakan token akses dengan klaim klien aplikasi; Amazon Cognito hanya mengembalikan nilai untuk atribut yang dapat dibaca klien aplikasi Anda. Token ID pengguna Anda dari aplikasi hanya berisi klaim yang sesuai dengan atribut yang dapat dibaca. Semua klien aplikasi dapat menulis atribut yang diperlukan kumpulan pengguna. Anda hanya dapat menyetel nilai atribut dalam permintaan API kumpulan pengguna Amazon Cognito jika Anda juga memberikan nilai untuk atribut wajib yang belum memiliki nilai.

Atribut khusus memiliki fitur berbeda untuk izin baca dan tulis. Anda dapat membuatnya sebagai mutable atau immutable untuk kumpulan pengguna, dan Anda dapat mengaturnya sebagai atribut baca atau tulis untuk klien aplikasi apa pun.

Atribut kustom yang tidak dapat diubah dapat diperbarui sekali, selama pembuatan pengguna. Anda dapat mengisi atribut yang tidak dapat diubah dengan metode berikut.

- `SignUp`: Pengguna mendaftar dengan klien aplikasi yang memiliki akses tulis ke atribut kustom yang tidak dapat diubah. Mereka memberikan nilai untuk atribut itu.
- `Masuk dengan iDP pihak ketiga`: Pengguna masuk ke klien aplikasi yang memiliki akses tulis ke atribut kustom yang tidak dapat diubah. Konfigurasi kumpulan pengguna Anda untuk idP mereka memiliki aturan untuk memetakan klaim yang diberikan ke atribut yang tidak dapat diubah.
- `AdminCreateUser`: Anda memberikan nilai untuk atribut yang tidak dapat diubah.

Untuk informasi tentang cakupan yang dapat Anda tetapkan ke klien aplikasi, lihat. [Cakupan, M2M, dan otorisasi API dengan server sumber daya](#)

Anda dapat mengubah izin dan cakupan atribut setelah Anda telah membuat kolam pengguna Anda.

## Menambahkan persyaratan kata sandi kumpulan pengguna

Kata sandi yang kuat dan kompleks adalah praktik terbaik keamanan untuk kumpulan pengguna Anda. Terutama dalam aplikasi yang terbuka untuk internet, kata sandi yang lemah dapat mengekspos kredensi pengguna Anda ke sistem yang menebak kata sandi dan mencoba mengakses data Anda. Semakin kompleks kata sandi, semakin sulit untuk ditebak. Amazon Cognito memiliki alat tambahan untuk administrator yang sadar keamanan, seperti [fitur keamanan tingkat lanjut](#) dan

[ACL AWS WAF web](#), tetapi kebijakan kata sandi Anda adalah elemen sentral dari keamanan direktori pengguna Anda.

Kata sandi untuk pengguna lokal di kumpulan pengguna Amazon Cognito tidak kedaluwarsa secara otomatis. Sebagai praktik terbaik, catat waktu, tanggal, dan metadata pengaturan ulang kata sandi pengguna di sistem eksternal. Dengan log eksternal usia kata sandi, aplikasi Anda atau pemicu Lambda dapat mencari usia kata sandi pengguna dan memerlukan pengaturan ulang setelah periode tertentu.

Anda dapat mengonfigurasi kumpulan pengguna agar memerlukan kompleksitas kata sandi minimum yang sesuai dengan standar keamanan Anda. Kata sandi yang kompleks memiliki panjang minimal delapan karakter. Mereka juga termasuk campuran huruf besar, numerik, dan karakter khusus.

Untuk menyetel kebijakan kata sandi kumpulan pengguna

1. Buat kumpulan pengguna dan arahkan ke langkah Konfigurasi persyaratan keamanan, atau akses kumpulan pengguna yang ada dan arahkan ke tab Pengalaman masuk.
2. Arahkan ke kebijakan Kata Sandi.
3. Pilih mode kebijakan Kata Sandi. Cognito default mengonfigurasi kumpulan pengguna Anda dengan pengaturan minimum yang disarankan. Anda juga dapat memilih kebijakan kata sandi khusus.
4. Tetapkan panjang minimum Kata Sandi. Semua pengguna harus mendaftar atau dibuat dengan kata sandi yang panjangnya lebih besar dari atau sama dengan nilai ini. Anda dapat mengatur nilai minimum ini setinggi 99, tetapi pengguna Anda dapat mengatur kata sandi hingga 256 karakter.
5. Konfigurasi aturan kompleksitas kata sandi di bawah Persyaratan Kata Sandi. Pilih jenis karakter—angka, karakter khusus, huruf besar, dan huruf kecil—yang ingin Anda perlukan setidaknya satu dari kata sandi setiap pengguna.

Anda dapat meminta setidaknya satu dari karakter berikut dalam kata sandi. Setelah Amazon Cognito memverifikasi bahwa kata sandi berisi karakter minimum yang diperlukan, kata sandi pengguna Anda dapat berisi karakter tambahan dari jenis apa pun hingga panjang kata sandi maksimum.

- [Huruf latin dasar huruf besar dan kecil](#)
- Nomor
- Karakter khusus berikut.

```
^ $ * . [ ] { } ( ) ? " ! @ # % & / \ , > < ' : ; | _ ~ ` = + -
```

- Karakter spasi non-terkemuka dan tidak tertinggal.
6. Tetapkan nilai untuk kata sandi sementara yang ditetapkan oleh administrator kedaluwarsa. Setelah periode ini berlalu, pengguna baru yang Anda buat di konsol Amazon Cognito atau dengan tidak `AdminCreateUser` dapat masuk dan menetapkan kata sandi baru. Setelah mereka masuk dengan kata sandi sementara, akun pengguna mereka tidak pernah kedaluwarsa. Untuk memperbarui durasi kata sandi di API kumpulan pengguna Amazon Cognito, tetapkan nilai untuk permintaan [TemporaryPasswordValidityDays](#) Anda [CreateUserPool](#) atau [UpdateUserPool](#) API.
- Untuk mengatur ulang akses untuk akun pengguna yang kedaluwarsa, lakukan salah satu hal berikut.
    - Hapus profil pengguna dan buat yang baru.
    - Tetapkan kata sandi permanen baru dalam permintaan [AdminSetUserPassword](#) API.
    - Buat kode konfirmasi baru dalam permintaan [AdminResetUserPassword](#) API.

## Pengaturan email untuk kumpulan pengguna Amazon Cognito

Peristiwa tertentu di aplikasi klien kumpulan pengguna Anda dapat menyebabkan Amazon Cognito mengirim email kepada pengguna Anda. Misalnya, jika Anda mengonfigurasi kolam pengguna agar memerlukan verifikasi email, Amazon Cognito akan mengirim email saat pengguna mendaftar akun baru di aplikasi Anda atau menyetel ulang kata sandi mereka. Tergantung pada tindakan yang memulai email, email berisi kode verifikasi atau kata sandi sementara.

Untuk menangani pengiriman email, Anda dapat menggunakan salah satu opsi berikut:

- [Konfigurasi email default](#) yang dibangun ke dalam layanan Amazon Cognito.
- [Konfigurasi Layanan Email Sederhana Amazon \(Amazon SES\)](#) Anda.

Anda dapat mengubah opsi pengiriman setelah membuat kumpulan pengguna.

Amazon Cognito mengirimkan pesan email ke pengguna Anda dengan kode yang dapat mereka masukkan atau tautan URL yang dapat mereka pilih. Tabel berikut menunjukkan peristiwa yang dapat menghasilkan pesan email.

## Opsi pesan

Aktifitas	Operasi API	Opsi pengiriman	Opsi format	Dapat disesuaikan	Template pesan
Lupa kata sandi	<a href="#">ForgotPassword</a>	Email, SMS	code	Tidak	N/A
Undangan	<a href="#">AdminCreateUser</a>	Email, SMS	code	Ya	Pesan undangan
Registrasi mandiri	<a href="#">SignUp</a>	Email, SMS	kode, tautan	Ya	Pesan verifikasi
Alamat email atau verifikasi nomor telepon	<a href="#">UpdateUserAttributes</a>	Email, SMS	code	Ya	Pesan verifikasi
Autentikasi multi-faktor (MFA)	<a href="#">AdminInitiateAuth</a> , <a href="#">InitiateAuth</a>	SMS	code	Ya <sup>1</sup>	Pesan MFA

<sup>1</sup> Untuk pesan SMS.

Amazon SES mengenakan biaya untuk pesan email. Untuk informasi lebih lanjut, lihat [Harga Amazon SES](#).

## Konfigurasi email default

Amazon Cognito dapat menggunakan konfigurasi email defaultnya untuk menangani pengiriman email untuk Anda. Saat Anda menggunakan opsi default, Amazon Cognito membatasi jumlah email yang dikirim setiap hari untuk kumpulan pengguna Anda. Untuk informasi tentang batas layanan, lihat [Kuota di Amazon Cognito](#). Untuk lingkungan produksi biasa, batas email default di bawah volume pengiriman yang diperlukan. Untuk mengaktifkan volume pengiriman yang lebih tinggi, Anda dapat menggunakan konfigurasi email Amazon SES.

Bila Anda menggunakan konfigurasi default, Anda menggunakan sumber daya Amazon SES yang dikelola oleh AWS untuk mengirim pesan email. [Amazon SES menambahkan alamat email](#)



[yang mengembalikan pantulan keras ke daftar penindasan tingkat akun atau daftar penindasan global](#). Jika alamat email yang tidak terkirim menjadi terkirim nanti, Anda tidak dapat mengontrol penghapusannya dari daftar penekanan saat kumpulan pengguna dikonfigurasi untuk menggunakan konfigurasi default. Alamat email dapat tetap berada di daftar penindasan AWS yang dikelola tanpa batas waktu. Untuk mengelola alamat email yang tidak terkirim, gunakan konfigurasi email Amazon SES Anda dengan daftar penindasan tingkat akun, seperti yang dijelaskan di bagian berikutnya.

Bila Anda menggunakan konfigurasi email default, Anda dapat menggunakan salah satu dari alamat email berikut sebagai alamat FROM:

- Alamat email default, [no-reply@verificationemail.com](mailto:no-reply@verificationemail.com).
- Alamat email khusus. Sebelum Anda dapat menggunakan alamat email Anda sendiri, Anda harus memverifikasinya dengan Amazon SES dan memberikan izin Amazon Cognito untuk menggunakan alamat ini.

## Konfigurasi email Amazon SES

Aplikasi Anda mungkin memerlukan volume pengiriman yang lebih tinggi daripada yang tersedia dengan opsi default. Untuk meningkatkan kemungkinan volume pengiriman, gunakan sumber daya Amazon SES Anda dengan kumpulan pengguna Anda untuk mengirim email kepada pengguna Anda. Anda juga dapat [memantau aktivitas pengiriman email](#) saat mengirim pesan email dengan konfigurasi Amazon SES Anda sendiri.

Sebelum Anda dapat menggunakan konfigurasi Amazon SES Anda, Anda harus memverifikasi satu atau beberapa alamat email, atau domain, dengan Amazon SES. Gunakan alamat email terverifikasi, atau alamat dari domain terverifikasi, sebagai alamat email FROM yang Anda tetapkan ke kumpulan pengguna Anda. Saat Amazon Cognito mengirim email ke pengguna, Amazon SES memanggil Amazon SES untuk Anda dan menggunakan alamat email Anda.

Saat Anda menggunakan konfigurasi Amazon SES, ketentuan berikut berlaku:

- Batas pengiriman email untuk kumpulan pengguna Anda adalah batas yang sama yang berlaku untuk alamat email terverifikasi Amazon SES Anda di alamat email Anda Akun AWS.
- [Anda dapat mengelola pesan ke alamat email yang tidak terkirim dengan daftar penindasan tingkat akun di Amazon SES yang mengesampingkan daftar penindasan global](#). Saat Anda menggunakan daftar penindasan tingkat akun, pantulan pesan email memengaruhi reputasi akun Anda sebagai pengirim. Untuk informasi selengkapnya, lihat [Menggunakan daftar penekanan tingkat akun Amazon SES di Panduan Pengembang](#) Layanan Email Sederhana Amazon.

## Wilayah konfigurasi email Amazon SES

Wilayah AWS Tempat Anda membuat kumpulan pengguna akan memiliki salah satu dari tiga persyaratan untuk konfigurasi pesan email dengan Amazon SES. Anda dapat mengirim pesan email dari Amazon SES di Wilayah yang sama dengan kumpulan pengguna Anda, beberapa Wilayah termasuk Wilayah yang sama, atau satu atau beberapa Wilayah terpencil. Untuk kinerja terbaik, kirim pesan email dengan identitas terverifikasi Amazon SES di Wilayah yang sama dengan kumpulan pengguna jika Anda memiliki opsi.

### Kategori persyaratan Wilayah untuk identitas terverifikasi Amazon SES

#### Hanya di Wilayah

Kumpulan pengguna Anda dapat mengirim pesan email dengan identitas terverifikasi Wilayah AWS sama dengan kumpulan pengguna. Dalam konfigurasi email default tanpa alamat FROM email khusus, Amazon Cognito menggunakan identitas `no-reply@verificationemail.com` terverifikasi di Wilayah yang sama.

#### Kompatibel ke belakang

Kumpulan pengguna Anda dapat mengirim pesan email dengan identitas terverifikasi di wilayah yang sama Wilayah AWS atau di salah satu Wilayah alternatif berikut:

- AS Timur (Virginia Utara)
- US West (Oregon)
- Eropa (Irlandia)

Fitur ini mendukung kontinuitas sumber daya kumpulan pengguna yang mungkin telah Anda buat agar sesuai dengan persyaratan Amazon Cognito saat layanan diluncurkan. Kumpulan pengguna dari periode itu hanya dapat mengirim pesan email dengan identitas terverifikasi dalam jumlah Wilayah AWS terbatas. Dalam konfigurasi email default tanpa alamat FROM email khusus, Amazon Cognito menggunakan identitas `no-reply@verificationemail.com` terverifikasi di Wilayah yang sama.

#### Wilayah Alternatif

Kumpulan pengguna Anda dapat mengirim pesan email dengan identitas terverifikasi di alternatif Wilayah AWS yang berada di luar Wilayah kumpulan pengguna. Konfigurasi ini terjadi ketika Amazon SES tidak tersedia di Wilayah tempat Amazon Cognito tersedia.

Kebijakan otorisasi pengiriman Amazon SES untuk identitas terverifikasi Anda di Wilayah alternatif harus mempercayai prinsip layanan Amazon Cognito dari Wilayah asal. Untuk informasi selengkapnya, lihat [Untuk memberikan izin untuk menggunakan konfigurasi email default](#).

Di beberapa Wilayah ini, Amazon Cognito membagi pesan email antara dua Wilayah alternatif untuk konfigurasi email default. COGNITO\_DEFAULT Dalam kasus ini, untuk menggunakan alamat FROM email khusus, kebijakan otorisasi pengiriman Amazon SES untuk identitas terverifikasi Anda di setiap Wilayah alternatif harus mempercayai prinsip layanan Amazon Cognito dari Wilayah asal. Untuk informasi selengkapnya, lihat [Untuk memberikan izin untuk menggunakan konfigurasi email default](#). Dengan konfigurasi email Amazon SES DEVELOPER di Wilayah ini, Anda harus menggunakan identitas terverifikasi di Wilayah yang terdaftar pertama dan mengonfigurasinya untuk mempercayai prinsip layanan Amazon Cognito di Wilayah kumpulan pengguna. Misalnya, di kumpulan pengguna di Timur Tengah (UEA), konfigurasi identitas terverifikasi di Eropa (Frankfurt) untuk dipercayacognito-idp.me-central-1.amazonaws.com. Dalam konfigurasi email default tanpa alamat FROM email khusus, Amazon Cognito menggunakan identitas no-reply@verificationemail.com terverifikasi di setiap Wilayah.

#### Note

Di bawah kombinasi kondisi berikut, Anda harus menentukan SourceArn parameter [EmailConfiguration](#) dengan wildcard di elemen Region, dalam format arn: `arn:aws:ses:region:account:identity/identity-name`. Ini memungkinkan kumpulan pengguna Anda untuk mengirim pesan email dengan identitas terverifikasi yang identik Akun AWS di keduanya. Wilayah AWS

- Anda EmailSendingAccount adalah COGNITO\_DEFAULT.
- Anda ingin menggunakan FROM alamat khusus.
- Kumpulan pengguna Anda mengirim email di Wilayah Alternatif.
- Kumpulan pengguna Anda memiliki Wilayah <sup>1</sup>Alternatif kedua yang ditentukan dalam tabel Wilayah yang didukung Amazon SES yang mengikuti.

Jika Anda membuat kumpulan pengguna secara terprogram—dengan SDK AWS, Amazon Cognito API, atau CLI, AWS CDK atau —kumpulan pengguna AWS CloudFormation Anda akan mengirimkan pesan email dengan identitas Amazon SES yang ditentukan parameter untuk kumpulan pengguna Anda. SourceArn [EmailConfiguration](#) Identitas Amazon SES harus menempati

dukungan Wilayah AWS. Jika Anda EmailSendingAccount COGNITO\_DEFAULT dan Anda tidak menentukan SourceArn parameter, Amazon Cognito mengirimkan pesan email dari `no-reply@verificationemail.com` menggunakan sumber daya di Wilayah tempat Anda membuat kumpulan pengguna.

Tabel berikut menunjukkan di Wilayah AWS mana Anda dapat menggunakan identitas Amazon SES dengan Amazon Cognito.

User pool Region	Opsi wilayah	Wilayah yang didukung Amazon SES
AS Timur (Virginia Utara)	Kompatibel ke belakang	AS Timur (Virginia N.), AS Barat (Oregon), Eropa (Irlandia)
AS Timur (Ohio)	Kompatibel ke belakang	AS Timur (Ohio), AS Timur (Virginia N.), AS Barat (Oregon), Eropa (Irlandia)
AS Barat (California Utara)	Hanya di Wilayah	AS Barat (California Utara)
AS Barat (Oregon)	Kompatibel ke belakang	AS Timur (Virginia N.), AS Barat (Oregon), Eropa (Irlandia)
Kanada (Pusat)	Kompatibel ke belakang	Kanada (Tengah), AS Timur (Virginia N.), AS Barat (Oregon), Eropa (Irlandia)
Asia Pasifik (Tokyo)	Kompatibel ke belakang	Asia Pasifik (Tokyo), AS Timur (Virginia N.), AS Barat (Oregon), Eropa (Irlandia)
Asia Pasifik (Seoul)	Kompatibel ke belakang	Asia Pasifik (Seoul), AS Timur (Virginia N.), AS Barat (Oregon), Eropa (Irlandia)

User pool Region	Opsi wilayah	Wilayah yang didukung Amazon SES
Asia Pasifik (Mumbai)	Kompatibel ke belakang	Asia Pasifik (Mumbai), AS Timur (Virginia N.), AS Barat (Oregon), Eropa (Irlandia)
Asia Pasifik (Hyderabad)	Wilayah Alternatif	Asia Pasifik (Mumbai), Asia Pasifik (Singapura) <sup>1</sup>
Asia Pasifik (Singapura)	Kompatibel ke belakang	Asia Pasifik (Singapura), AS Timur (Virginia N.), AS Barat (Oregon), Eropa (Irlandia)
Asia Pasifik (Sydney)	Kompatibel ke belakang	Asia Pasifik (Sydney), AS Timur (Virginia N.), AS Barat (Oregon), Eropa (Irlandia)
Asia Pasifik (Osaka)	Hanya di Wilayah	Asia Pasifik (Osaka)
Asia Pasifik (Jakarta)	Hanya di Wilayah	Asia Pasifik (Jakarta)
Asia Pasifik (Melbourne)	Wilayah Alternatif	Asia Pasifik (Sydney), Asia Pasifik (Singapura) <sup>1</sup>
Eropa (Irlandia)	Kompatibel ke belakang	AS Timur (Virginia N.), AS Barat (Oregon), Eropa (Irlandia)
Eropa (London)	Kompatibel ke belakang	Eropa (London), AS Timur (Virginia N.), AS Barat (Oregon), Eropa (Irlandia)
Eropa (Paris)	Hanya di Wilayah	Eropa (Paris)
Eropa (Frankfurt)	Kompatibel ke belakang	Eropa (Frankfurt), AS Timur (Virginia N.), AS Barat (Oregon), Eropa (Irlandia)

User pool Region	Opsi wilayah	Wilayah yang didukung Amazon SES
Eropa (Zürich)	Wilayah Alternatif	Eropa (Frankfurt), Eropa (London) <sup>1</sup>
Eropa (Stockholm)	Hanya di Wilayah	Eropa (Stockholm)
Eropa (Milan)	Hanya di Wilayah	Eropa (Milan)
Eropa (Spanyol)	Wilayah Alternatif	Eropa (Paris), Eropa (Stockholm) <sup>1</sup>
Timur Tengah (Bahrain)	Hanya di Wilayah	Timur Tengah (Bahrain)
Middle East (UAE)	Wilayah Alternatif	Eropa (Frankfurt), Eropa (London) <sup>1</sup>
Amerika Selatan (Sao Paulo)	Hanya di Wilayah	Amerika Selatan (Sao Paulo)
Israel (Tel Aviv)	Hanya di Wilayah	Israel (Tel Aviv)
Afrika (Cape Town)	Hanya di Wilayah	Afrika (Cape Town)

<sup>1</sup> Digunakan dalam kumpulan pengguna dengan konfigurasi email default. Amazon Cognito mendistribusikan pesan email di antara identitas terverifikasi dengan alamat email yang sama di setiap Wilayah. Untuk menggunakan FROM alamat khusus, konfigurasi EmailConfiguration dengan SourceArn parameter dalam formatarn: `arn:ses:*:${Account}:identity/${IdentityName}`.

## Mengkonfigurasi email untuk kumpulan pengguna Anda

Selesaikan langkah-langkah berikut untuk mengonfigurasi pengaturan email untuk kolam pengguna Anda. Bergantung pada pengaturan yang Anda gunakan, Anda mungkin memerlukan izin IAM di Amazon SES, AWS Identity and Access Management (IAM), dan Amazon Cognito.

**Note**

Anda tidak dapat membagikan sumber daya yang Anda buat dalam langkah-langkah ini Akun AWS. Misalnya, Anda tidak dapat mengonfigurasi kumpulan pengguna dalam satu akun, lalu menggunakannya dengan alamat email Amazon SES di akun lain. Jika Anda menggunakan Amazon Cognito di beberapa akun, ulangi langkah-langkah ini untuk setiap akun.

## Langkah 1: Verifikasi alamat email atau domain Anda dengan Amazon SES

Sebelum mengonfigurasi kumpulan pengguna, Anda harus memverifikasi satu atau beberapa domain atau alamat email dengan Amazon SES jika Anda ingin melakukan salah satu dari hal berikut:

- Gunakan alamat email Anda sendiri sebagai alamat FROM
- Gunakan konfigurasi Amazon SES Anda untuk menangani pengiriman email

Dengan memverifikasi alamat email atau domain Anda, Anda mengonfirmasi bahwa Anda memilikinya, yang membantu mencegah penggunaan yang tidak sah.

Untuk informasi tentang memverifikasi alamat email dengan Amazon SES, lihat [Memverifikasi Alamat Email](#) di Panduan Pengembang Layanan Email Sederhana Amazon. Untuk informasi tentang memverifikasi domain dengan Amazon SES, lihat [Memverifikasi domain](#).

## Langkah 2: Pindahkan akun Anda dari kotak pasir Amazon SES

Abaikan langkah ini jika Anda menggunakan konfigurasi email Amazon Cognito default.

Saat Anda pertama kali menggunakan Amazon SES di mana pun Wilayah AWS, itu menempatkan Anda Akun AWS di kotak pasir Amazon SES untuk Wilayah itu. Amazon SES menggunakan sandbox untuk mencegah penipuan dan penyalahgunaan. Jika Anda menggunakan konfigurasi Amazon SES untuk menangani pengiriman email, Anda harus Akun AWS keluar dari kotak pasir sebelum Amazon Cognito dapat mengirim email kepada pengguna Anda.

Dalam sandbox, Amazon SES memberlakukan pembatasan pada berapa banyak email Anda dapat mengirim dan di mana Anda dapat mengirim email tersebut. Anda dapat mengirim email hanya ke alamat dan domain yang telah Anda verifikasi dengan Amazon SES, atau Anda dapat mengirim email ke alamat simulator kotak surat Amazon SES. Saat Anda Akun AWS tetap berada di kotak pasir, jangan gunakan konfigurasi Amazon SES Anda untuk aplikasi yang sedang diproduksi. Dalam situasi ini, Amazon Cognito tidak dapat mengirim pesan ke alamat email pengguna Anda.

Untuk menghapus kotak pasir Anda Akun AWS dari kotak pasir, lihat [Berpindah dari kotak pasir Amazon SES di Panduan](#) Pengembang Layanan Email Sederhana Amazon.

### Langkah 3: Berikan izin email ke Amazon Cognito

Anda mungkin perlu memberikan izin khusus ke Amazon Cognito sebelum dapat mengirim email ke pengguna Anda. Izin yang Anda berikan, dan proses yang Anda gunakan untuk memberikannya, bergantung pada apakah Anda menggunakan konfigurasi email default, atau konfigurasi Amazon SES Anda.

Untuk memberikan izin untuk menggunakan konfigurasi email default

Selesaikan langkah ini hanya jika Anda mengonfigurasi kumpulan pengguna Anda ke Kirim email dengan Cognito atau atur `EmailSendingAccount` ke `COGNITO_DEFAULT`

Dengan konfigurasi email default, kumpulan pengguna Anda dapat mengirim pesan email dengan salah satu alamat berikut.

- Alamat default `no-reply@verificationemail.com`.
- Alamat FROM kustom dari alamat email atau domain terverifikasi Anda di Amazon SES.

Jika Anda menggunakan alamat khusus, Amazon Cognito memerlukan izin tambahan untuk mengirim email kepada pengguna dari alamat tersebut. Izin ini diberikan oleh [kebijakan otorisasi pengiriman](#) untuk alamat atau domain di Amazon SES. Jika Anda menggunakan konsol Amazon Cognito untuk menambahkan alamat khusus ke kumpulan pengguna, kebijakan akan secara otomatis dilampirkan ke alamat email terverifikasi Amazon SES. Namun, jika Anda mengonfigurasi kumpulan pengguna di luar konsol, seperti menggunakan AWS CLI atau Amazon Cognito API, Anda harus melampirkan kebijakan menggunakan [konsol Amazon SES](#) atau API. [PutIdentityPolicy](#)

#### Note

Anda hanya dapat mengonfigurasi alamat FROM di domain terverifikasi menggunakan API Amazon Cognito AWS CLI atau Amazon.

Kebijakan otorisasi pengiriman mengizinkan atau menolak akses berdasarkan sumber daya akun yang menggunakan Amazon Cognito untuk memanggil Amazon SES. [Untuk informasi selengkapnya tentang kebijakan berbasis sumber daya, lihat Panduan Pengguna IAM](#). Anda juga dapat menemukan contoh kebijakan berbasis sumber daya di Panduan Pengembang [Amazon](#) SES.



## Example Mengirim kebijakan otorisasi

Contoh berikut mengirim kebijakan otorisasi memberi Amazon Cognito kemampuan terbatas untuk menggunakan identitas terverifikasi Amazon SES. Amazon Cognito hanya dapat mengirim pesan email ketika melakukannya atas nama kumpulan pengguna dalam `aws:SourceArn` kondisi dan akun dalam kondisi tersebut `aws:SourceAccount`.

### Regions with Amazon SES

Kebijakan otorisasi pengiriman Anda di wilayah kumpulan pengguna atau Wilayah alternatif harus mengizinkan kepala layanan Amazon Cognito untuk mengirim pesan email. Lihat [tabel Wilayah](#) untuk informasi lebih lanjut. Jika Wilayah Kumpulan Pengguna Anda cocok dengan setidaknya satu nilai di Wilayah Amazon SES, konfigurasi kebijakan otorisasi pengiriman Anda dengan prinsipal layanan global dalam contoh berikut.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "stmnt1234567891234",
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "email.cognito-idp.amazonaws.com"
        ]
      },
      "Action": [
        "SES:SendEmail",
        "SES:SendRawEmail"
      ],
      "Resource": "<your SES identity ARN>",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "<your account number>"
        },
        "ArnLike": {
          "aws:SourceArn": "<your user pool ARN>"
        }
      }
    }
  ]
}
```

## Opt-in Regions without Amazon SES

Amazon SES tidak tersedia di semua keikutsertaan Wilayah AWS di mana Amazon Cognito tersedia. Timur Tengah (UEA) adalah contohnya, dan hanya dapat mengirim email dengan identitas terverifikasi di Eropa (Frankfurt) (`eu-central-1`). Di kumpulan pengguna dengan konfigurasi email default, Amazon Cognito juga mengirim pesan email dengan identitas terverifikasi di masing-masing dari dua Wilayah. Dalam kasus Timur Tengah (UEA), Wilayah tambahan adalah Eropa (London). Anda harus memperbarui kebijakan otorisasi pengiriman di kedua Wilayah.

Kebijakan otorisasi pengiriman Anda di setiap Wilayah alternatif harus mengizinkan kepala layanan Amazon Cognito di Wilayah keikutsertaan kumpulan pengguna untuk mengirim pesan email. Lihat [tabel Wilayah](#) untuk informasi lebih lanjut. Jika Wilayah Anda ditandai sebagai Wilayah Alternatif, konfigurasi kebijakan otorisasi pengiriman Anda dengan kepala layanan Regional seperti pada contoh berikut. Ganti contoh Region identifier `me-central-1` dengan Region ID yang diperlukan sesuai kebutuhan.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "cognito-idp.me-central-1.amazonaws.com"
        ]
      },
      "Action": [
        "SES:SendEmail",
        "SES:SendRawEmail"
      ],
      "Resource": "<your SES identity ARN>",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "<your account number>"
        },
        "ArnLike": {
          "aws:SourceArn": "<your user pool ARN>"
        }
      }
    }
  ]
}
```

```
}
```

Untuk informasi selengkapnya tentang sintaks kebijakan, lihat [Amazon SES mengirim kebijakan otorisasi](#) di Panduan Pengembang Layanan Email Sederhana Amazon.

Untuk contoh selengkapnya, lihat [Amazon SES mengirimkan contoh kebijakan otorisasi](#) di Panduan Pengembang Layanan Email Sederhana Amazon.

Untuk memberikan izin untuk menggunakan konfigurasi Amazon SES Anda

Jika Anda mengonfigurasi kolam pengguna Anda untuk menggunakan konfigurasi Amazon SES, Amazon Cognito memerlukan izin tambahan untuk memanggil Amazon SES atas nama Anda ketika mengirimkan email ke pengguna Anda. Otorisasi ini diberikan dengan layanan IAM.

Saat Anda mengonfigurasi kumpulan pengguna dengan opsi ini, Amazon Cognito membuat peran terkait layanan, yang merupakan jenis peran IAM, dalam peran Anda. Akun AWS Peran ini berisi izin yang memungkinkan Amazon Cognito mengakses Amazon SES dan mengirim email dengan alamat Anda.

Amazon Cognito membuat peran terkait layanan Anda dengan AWS kredensial sesi pengguna yang menyetel konfigurasi. Izin IAM sesi ini harus menyertakan tindakan. `iam:CreateServiceLinkedRole` Untuk informasi selengkapnya tentang izin di IAM, lihat [Manajemen akses untuk AWS sumber daya](#) di Panduan Pengguna IAM.

Untuk informasi lebih lanjut tentang peran yang terkait dengan layanan yang dibuat Amazon Cognito, lihat [Menggunakan peran terkait layanan untuk Amazon Cognito](#).

#### Langkah 4: Konfigurasi kumpulan pengguna Anda

Selesaikan langkah-langkah berikut jika Anda ingin mengonfigurasi kolam pengguna Anda dengan salah satu dari berikut ini:

- Alamat FROM kustom yang muncul sebagai pengirim email
- Alamat REPLY-TO kustom yang menerima pesan yang dikirim pengguna ke alamat FROM Anda
- Konfigurasi Amazon SES Anda

**Note**

Jika identitas terverifikasi Anda adalah alamat email, Amazon Cognito menetapkan alamat email tersebut sebagai alamat email FROM dan REPLY-TO secara default. Namun, jika identitas terverifikasi Anda adalah domain, Anda harus memberikan nilai untuk alamat email FROM dan REPLY-TO. Misalnya, jika domain terverifikasi Anda adalah example.com, Anda dapat menetapkan no-reply@example.com sebagai alamat email FROM dan REPLY-TO.

Abaikan prosedur ini jika Anda ingin menggunakan konfigurasi dan alamat email Amazon Cognito default.

Untuk mengonfigurasi kumpulan pengguna Anda untuk menggunakan alamat email khusus

1. Masuk ke [Konsol Amazon Cognito](#). Jika diminta, masukkan AWS kredensial Anda.
2. Pilih Kolam Pengguna.
3. Pilih kumpulan pengguna yang ada dari daftar.
4. Pilih tab Pesan, cari konfigurasi Email, pilih Edit.
5. Pada halaman Edit konfigurasi email, pilih Kirim email dari Amazon SES atau Kirim email dengan Amazon Cognito. Anda dapat menyesuaikan Wilayah SES, Set Konfigurasi, dan FROM nama pengirim hanya jika Anda memilih Kirim email dari Amazon SES.
6. Untuk menggunakan alamat FROM kustom, selesaikan langkah-langkah berikut:
  - a. Di bawah Wilayah SES, pilih Wilayah yang berisi alamat email terverifikasi Anda.
  - b. Di bawah dari alamat email, pilih alamat email Anda. Gunakan alamat email yang telah Anda verifikasi dengan Amazon SES.
  - c. (Opsional) Di bawah set Konfigurasi, pilih set konfigurasi untuk Amazon SES untuk digunakan. Membuat dan menyimpan perubahan ini menciptakan peran terkait layanan.
  - d. (Opsional) Di bawah alamat pengirim DARI, masukkan alamat email. Anda hanya dapat memberikan alamat email, atau alamat email dan nama ramah dalam format Jane Doe <janedoe@example.com>.
  - e. (Opsional) Di bawah alamat email REPLY-TO, masukkan alamat email tempat Anda ingin menerima pesan yang dikirim pengguna ke alamat FROM Anda.
7. Pilih Simpan perubahan.

## Topik Terkait

- [Menyesuaikan pesan verifikasi email](#)
- [Menyesuaikan pesan undangan pengguna](#)

## Pengaturan pesan SMS untuk kolam pengguna Amazon Cognito

Beberapa peristiwa Amazon Cognito untuk kolam pengguna Anda dapat menyebabkan Amazon Cognito mengirim pesan teks SMS ke pengguna Anda. Misalnya, jika Anda mengonfigurasi kolam pengguna agar memerlukan verifikasi telepon, Amazon Cognito akan mengirim pesan teks SMS saat pengguna mendaftar akun baru di aplikasi Anda atau menyetel ulang kata sandi mereka. Tergantung pada tindakan yang memulai pesan teks SMS, pesan berisi kode verifikasi, sandi sementara, atau pesan pembuka.

Amazon Cognito menggunakan Amazon Simple Notification Service (Amazon SNS) untuk pengiriman pesan teks SMS. Jika Anda mengirim pesan teks melalui Amazon Cognito atau Amazon SNS untuk pertama kalinya, Amazon SNS menempatkan Anda di lingkungan kotak pasir. Di lingkungan kotak pasir, Anda dapat menguji aplikasi Anda untuk pesan teks SMS. Di sandbox, pesan hanya bisa dikirim ke nomor telepon terverifikasi.

Amazon SNS mengenakan biaya untuk pesan teks SMS. Untuk informasi selengkapnya, lihat [harga Amazon SNS](#).

### Note

Karena volume lalu lintas SMS yang tidak diminta di seluruh dunia, beberapa pemerintah memberlakukan hambatan antara pengirim dan penerima pesan SMS. Saat Anda menggunakan pesan SMS untuk MFA dan pembaruan pengguna, Anda harus mengambil langkah-langkah tambahan untuk memastikan bahwa pesan Anda terkirim. Anda juga harus memantau peraturan terkait pesan SMS di negara-negara tempat pengguna Anda mungkin tinggal dan menjaga konfigurasi pesan SMS Anda tetap terkini. Untuk informasi selengkapnya, lihat [Pesan teks seluler \(SMS\)](#) di Panduan Pengembang Layanan Pemberitahuan Sederhana Amazon.

Penggunaan pesan SMS untuk mengautentikasi dan memverifikasi pengguna bukanlah praktik terbaik keamanan. Nomor telepon dapat mengubah pemilik, dan mungkin tidak secara andal mewakili sesuatu yang Anda miliki faktor MFA untuk pengguna Anda. Sebagai gantinya, terapkan TOTP MFA di aplikasi Anda atau dengan iDP pihak ketiga Anda. Anda

juga dapat membuat faktor otentikasi kustom tambahan dengan [Tantangan otentikasi kustom pemicu Lambda](#).

Amazon Cognito mengirimkan pesan SMS ke pengguna Anda dengan kode yang dapat mereka masukkan. Tabel berikut menunjukkan peristiwa yang dapat menghasilkan pesan SMS.

Opsi pesan

Aktifitas	Operasi API	Opsi pengiriman	Opsi format	Dapat disesuaikan	Template pesan
Lupa kata sandi	<a href="#">ForgotPassword</a>	Email, SMS	code	Tidak	N/A
Undangan	<a href="#">AdminCreateUser</a>	Email, SMS	code	Ya	Pesan undangan
Registrasi mandiri	<a href="#">SignUp</a>	Email, SMS	kode, tautan	Ya	Pesan verifikasi
Alamat email atau verifikasi nomor telepon	<a href="#">UpdateUserAttributes</a>	Email, SMS	code	Ya	Pesan verifikasi
Autentikasi multi-faktor (MFA)	<a href="#">AdminInitiateAuth</a> , <a href="#">InitiateAuth</a>	SMS, aplikasi otentikator	code	Ya <sup>1</sup>	Pesan MFA

<sup>1</sup> Untuk pesan SMS.

## Menyiapkan pesan SMS untuk pertama kalinya di kumpulan pengguna Amazon Cognito

Amazon Cognito menggunakan Amazon SNS untuk mengirim pesan SMS ke kolam pengguna Anda. Anda juga dapat menggunakan a [Pemicu Lambda pengirim SMS kustom](#) untuk menggunakan sumber daya Anda sendiri untuk mengirim pesan SMS. Pertama kali Anda mengatur Amazon SNS

untuk mengirim pesan teks SMS secara khusus Wilayah AWS, Amazon SNS menempatkan Akun AWS Anda di kotak pasir SMS untuk Wilayah itu. Amazon SNS menggunakan kotak pasir untuk mencegah penipuan dan penyalahgunaan dan untuk memenuhi persyaratan kepatuhan. [Saat Anda Akun AWS berada di kotak pasir, Amazon SNS memberlakukan beberapa batasan](#). Misalnya, Anda dapat mengirim pesan teks ke maksimum 10 nomor telepon yang telah Anda verifikasi dengan Amazon SNS. Saat Anda Akun AWS tetap berada di kotak pasir, jangan gunakan konfigurasi Amazon SNS Anda untuk aplikasi yang sedang diproduksi. Saat Anda berada di sandbox, Amazon Cognito tidak dapat mengirim pesan ke nomor telepon pengguna Anda.

Untuk mengirim pesan teks SMS ke pengguna pool pengguna

1. [Siapkan peran IAM yang dapat digunakan Amazon Cognito untuk mengirim pesan SMS dengan Amazon SNS](#)
2. [Pilih pesan Wilayah AWS SMS Amazon SNS untuk](#)
3. [Dapatkan identitas originasi untuk mengirim pesan SMS ke nomor telepon AS](#)
4. [Konfirmasikan bahwa Anda berada di sandbox SMS](#)
5. [Pindahkan akun Anda keluar dari sandbox Amazon SNS](#)
6. [Verifikasi nomor telepon untuk Amazon Cognito di Amazon SNS](#)
7. [Selesaikan pengaturan kumpulan pengguna di Amazon Cognito](#)

## Siapkan peran IAM yang dapat digunakan Amazon Cognito untuk mengirim pesan SMS dengan Amazon SNS

Saat Anda mengirim pesan SMS dari kumpulan pengguna Anda, Amazon Cognito mengambil peran IAM di akun Anda. Amazon Cognito menggunakan `sns:Publish` izin yang ditetapkan untuk peran tersebut untuk mengirim pesan SMS ke pengguna Anda. Di konsol Amazon Cognito, Anda dapat mengatur pemilihan peran IAM dari tab Pesan kumpulan pengguna, di bawah SMS atau membuat pilihan ini selama panduan pembuatan kumpulan pengguna.

Contoh kebijakan kepercayaan peran IAM berikut memberikan kumpulan pengguna Amazon Cognito kemampuan terbatas untuk mengambil peran tersebut. Amazon Cognito hanya dapat mengambil peran ketika melakukannya atas nama kumpulan pengguna dalam `aws:SourceArn` kondisi dan kondisi. Akun AWS `aws:SourceAccount`

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

```
    "Effect": "Allow",
    "Principal": {
      "Service": "cognito-idp.amazonaws.com"
    },
    "Action": "sts:AssumeRole",
    "Condition": {
      "StringEquals": {
        "aws:SourceAccount": "<your account number>"
      },
      "ArnLike": {
        "aws:SourceArn": "<your user pool ARN>"
      }
    }
  }
}]
}
```

Anda dapat menentukan [ARN kumpulan pengguna](#) yang tepat atau ARN wildcard dalam nilai kondisi. `aws:SourceArn` Cari ARN kumpulan pengguna Anda di AWS Management Console atau dengan permintaan [DescribeUserPoolAPI](#).

Untuk informasi selengkapnya tentang peran IAM dan kebijakan kepercayaan, lihat [Istilah dan konsep peran](#) di Panduan AWS Identity and Access Management Pengguna.

## Pilih pesan Wilayah AWS SMS Amazon SNS untuk

Di beberapa Wilayah AWS, Anda dapat memilih Wilayah yang berisi sumber daya Amazon SNS yang ingin Anda gunakan untuk pesan SMS Amazon Cognito. Di Wilayah AWS mana pun Amazon Cognito tersedia, kecuali untuk Asia Pasifik (Seoul), Anda dapat menggunakan sumber daya Amazon SNS di Wilayah AWS tempat Anda membuat kumpulan pengguna. Untuk membuat pesan SMS Anda lebih cepat dan lebih andal ketika Anda memiliki pilihan Wilayah, gunakan sumber daya Amazon SNS di Wilayah yang sama dengan kumpulan pengguna Anda.

### Note

Di dalam AWS Management Console, Anda hanya dapat mengubah Wilayah untuk sumber daya SMS setelah Anda beralih ke pengalaman konsol Amazon Cognito yang baru.

Pilih Wilayah untuk sumber daya SMS di langkah Konfigurasi pengiriman pesan dari panduan kumpulan pengguna baru. Anda juga dapat memilih Edit di bawah SMS di tab Pesan dari kumpulan pengguna yang ada.



Saat peluncuran, bagi sebagian orang Wilayah AWS, Amazon Cognito mengirim pesan SMS dengan sumber daya Amazon SNS di Wilayah alternatif. Untuk mengatur Wilayah pilihan Anda, gunakan `SnsRegion` parameter [SmsConfigurationType](#) objek untuk kumpulan pengguna Anda. Saat Anda membuat sumber daya kumpulan pengguna Amazon Cognito secara terprogram di Wilayah Amazon Cognito dari tabel berikut dan Anda tidak memberikan **SnsRegion** parameter, kumpulan pengguna Anda dapat mengirim pesan SMS dengan sumber daya Amazon SNS di Wilayah Amazon SNS lama.

Kumpulan pengguna Amazon Cognito di Asia Pasifik (Seoul) Wilayah AWS harus menggunakan konfigurasi Amazon SNS Anda di Wilayah Asia Pasifik (Tokyo).

Amazon SNS menetapkan kuota pengeluaran untuk semua akun baru sebesar \$1,00 (USD) per bulan. Anda mungkin telah meningkatkan batas pengeluaran Anda dalam Wilayah AWS yang Anda gunakan dengan Amazon Cognito. Sebelum Anda mengubah pesan SMS Amazon SNS Wilayah AWS untuk Amazon, buka kasus peningkatan kuota di AWS Support Center untuk meningkatkan batas Anda di Wilayah baru. Untuk informasi selengkapnya, lihat [Meminta peningkatan kuota pengeluaran SMS bulanan Anda untuk Amazon SNS](#) di Panduan Pengembang Layanan Pemberitahuan Sederhana Amazon.

Anda dapat mengirim pesan SMS untuk Wilayah Amazon Cognito apa pun di tabel berikut dengan sumber daya Amazon SNS di Wilayah Amazon SNS yang sesuai.

Wilayah Amazon Cognito	Wilayah Amazon SNS
AS Timur (Ohio)	AS Timur (Ohio), AS Timur (Virginia N.)
Kanada (Pusat)	Kanada (Tengah), AS Timur (Virginia N.)
Eropa (Frankfurt)	Eropa (Frankfurt), Eropa (Irlandia)
Eropa (London)	Eropa (London), Eropa (Irlandia)
Asia Pasifik (Seoul)	Asia Pasifik (Tokyo)
AS Timur (Virginia Utara)	AS Timur (Virginia Utara)
AS Barat (California Utara)	AS Barat (California Utara)
AS Barat (Oregon)	AS Barat (Oregon)
Asia Pasifik (Mumbai)	Asia Pasifik (Mumbai), Asia Pasifik (Singapura)

Wilayah Amazon Cognito	Wilayah Amazon SNS
Asia Pasifik (Hyderabad)	Asia Pasifik (Hyderabad)
Asia Pasifik (Singapura)	Asia Pasifik (Singapura)
Asia Pasifik (Sydney)	Asia Pasifik (Sydney)
Asia Pasifik (Tokyo)	Asia Pasifik (Tokyo)
Asia Pasifik (Jakarta)	Asia Pasifik (Jakarta)
Asia Pasifik (Osaka)	Asia Pasifik (Osaka)
Asia Pasifik (Melbourne)	Asia Pasifik (Melbourne)
Eropa (Irlandia)	Eropa (Irlandia)
Eropa (Paris)	Eropa (Paris)
Eropa (Stockholm)	Eropa (Stockholm)
Eropa (Milan)	Eropa (Milan)
Eropa (Spanyol)	Eropa (Spanyol)
Timur Tengah (Bahrain)	Timur Tengah (Bahrain)
Amerika Selatan (Sao Paulo)	Amerika Selatan (Sao Paulo)
Israel (Tel Aviv)	Israel (Tel Aviv)
Afrika (Cape Town)	Afrika (Cape Town)
Timur Tengah (UEA)	Timur Tengah (UEA)
Eropa (Zürich)	Eropa (Zürich)

## Dapatkan identitas originasi untuk mengirim pesan SMS ke nomor telepon AS

Jika Anda berencana untuk mengirim pesan teks SMS ke nomor telepon AS, Anda harus mendapatkan identitas originasi, terlepas dari apakah Anda membangun lingkungan pengujian kotak pasir SMS, atau lingkungan produksi.

Mulai 1 Juni 2021, operator AS memerlukan identitas originasi untuk mengirim pesan ke nomor telepon AS. Jika Anda belum memiliki identitas originasi, Anda harus mendapatkannya. Untuk mempelajari cara mendapatkan identitas asal, lihat [Meminta nomor](#) dalam Panduan Pengguna Amazon Pinpoint.

Jika Anda beroperasi di bawah ini Wilayah AWS, Anda harus membuka AWS Support tiket untuk mendapatkan identitas originasi. Untuk instruksi, lihat [Meminta dukungan untuk pesan SMS](#) di Panduan Developer Amazon Simple Notification Service.

- AS Timur (Ohio)
- Eropa (Stockholm)
- Eropa (Paris)
- Eropa (Milan)
- Middle East (Bahrain)
- Amerika Selatan (São Paulo)
- AS Barat (California Utara)

Ketika Anda memiliki lebih dari satu identitas originasi yang sama Wilayah AWS, Amazon SNS memilih jenis identitas originasi dalam urutan prioritas berikut: kode pendek, 10DLC, nomor bebas pulsa. Anda tidak dapat mengubah prioritas ini. Untuk informasi selengkapnya, lihat [FAQ Amazon SNS](#).

## Konfirmasikan bahwa Anda berada di sandbox SMS

Gunakan prosedur berikut untuk mengonfirmasi bahwa Anda berada di kotak pasir SMS. Ulangi untuk setiap Wilayah AWS tempat Anda memiliki kumpulan pengguna Amazon Cognito produksi.

Tinjau status kotak pasir SMS di konsol Amazon Cognito

Untuk mengonfirmasi bahwa Anda berada di kotak pasir SMS

1. Masuk ke [Konsol Amazon Cognito](#). Jika diminta, masukkan AWS kredensial Anda.

2. Pilih Kolam Pengguna.
3. Pilih kumpulan pengguna yang ada dari daftar.
4. Pilih tab Pesan.
5. Di bagian konfigurasi SMS, perluas Pindah ke lingkungan produksi Amazon SNS. Jika akun Anda ada di kotak pasir SMS, Anda akan melihat pesan berikut:

```
You are currently in the SMS Sandbox and cannot send SMS messages to unverified numbers.
```

Jika Anda tidak melihat pesan ini, maka seseorang telah menyiapkan pesan SMS di akun Anda. Loncat ke [Selesaikan pengaturan kumpulan pengguna di Amazon Cognito](#).

6. Pilih tautan [Amazon SNS](#) di pesan. Ini membuka konsol Amazon SNS di tab baru.
7. Verifikasi bahwa Anda berada di lingkungan sandbox. Pesan konsol menunjukkan status kotak pasir Anda dan Wilayah AWS, sebagai berikut:

```
This account is in the SMS sandbox in US East (N. Virginia).
```

## Pindahkan akun Anda keluar dari sandbox Amazon SNS

Jika Anda menguji aplikasi dan hanya perlu mengirim pesan SMS ke nomor telepon yang dapat diverifikasi oleh administrator, lewati langkah ini.

Untuk menggunakan aplikasi Anda dalam produksi, pindahkan akun Anda dari kotak pasir SMS dan masuk ke produksi. Setelah Anda mengonfigurasi identitas originasi di Wilayah AWS yang berisi sumber daya Amazon SNS yang ingin Anda gunakan Amazon Cognito, Anda dapat memverifikasi nomor telepon AS saat Akun AWS Anda tetap berada di kotak pasir SMS. Saat lingkungan Amazon SNS Anda dalam produksi, Anda tidak perlu memverifikasi nomor telepon pengguna di Amazon SNS untuk mengirim pesan SMS ke pengguna Anda.

Untuk petunjuk mendetail, lihat [Keluar dari Kotak Pasir SMS](#) di Panduan Pengembang Layanan Pemberitahuan Sederhana Amazon.

## Verifikasi nomor telepon untuk Amazon Cognito di Amazon SNS

Jika Anda telah memindahkan akun Anda dari kotak pasir SMS, lewati langkah ini.

Saat Anda berada di kotak pasir SMS, Anda dapat mengirim pesan ke nomor telepon apa pun yang telah Anda verifikasi dengan Amazon SNS.

Untuk memverifikasi nomor telepon, lakukan hal berikut:

1. Tambahkan nomor telepon tujuan Sandbox di bagian Pesan teks (SMS) di konsol Amazon SNS.
2. Terima pesan SMS dengan kode di nomor telepon yang Anda berikan.
3. Masukkan kode Verifikasi dari pesan SMS di konsol Amazon SNS.

Untuk instruksi detail, lihat [Menambahkan dan memverifikasi nomor telepon di sandbox SMS](#) di Panduan Developer Amazon Simple Notification Service.

#### Note

Amazon SNS membatasi jumlah nomor telepon tujuan yang dapat Anda verifikasi saat Anda berada di kotak pasir SMS. Lihat [kotak pasir SMS](#) di Panduan Pengembang Layanan Pemberitahuan Sederhana Amazon.

## Selesaikan pengaturan kumpulan pengguna di Amazon Cognito

Kembali ke tab browser tempat Anda [membuat](#) atau [mengedit](#) kolam pengguna Anda. Selesaikan prosedur. Ketika Anda telah berhasil menambahkan konfigurasi SMS ke kumpulan pengguna Anda, Amazon Cognito mengirimkan pesan pengujian ke nomor telepon internal untuk memverifikasi bahwa konfigurasi Anda berfungsi. Amazon SNS mengenakan biaya untuk setiap pesan SMS tes.

## Menggunakan token dengan kolam pengguna

Autentikasi pengguna dan berikan akses ke sumber daya dengan token. Klaim dalam token adalah informasi tentang pengguna Anda. Token ID berisi klaim tentang identitas mereka, seperti nama pengguna, nama keluarga, dan alamat email mereka. Token akses berisi klaim seperti scope yang dapat digunakan pengguna yang diautentikasi untuk mengakses API pihak ketiga, operasi API layanan mandiri pengguna Amazon Cognito, dan file. [Titik akhir UserInfo](#) Token akses dan ID keduanya menyertakan `cognito:groups` klaim yang berisi keanggotaan grup pengguna Anda di kumpulan pengguna Anda. Untuk informasi selengkapnya tentang grup kumpulan pengguna, lihat [Menambahkan grup ke kumpulan pengguna](#).

Amazon Cognito juga memiliki token penyegaran yang dapat Anda gunakan untuk mendapatkan token baru atau mencabut token yang ada. [Refresh token](#) untuk mengambil ID baru dan token akses. [Cabut token untuk mencabut](#) akses pengguna yang diizinkan oleh token penyegaran.

Amazon Cognito mengeluarkan token sebagai string yang dikodekan Base64. Anda dapat memecahkan kode ID Amazon Cognito atau token akses apa pun dari base64 ke JSON teks biasa. Token penyegaran Amazon Cognito dienkripsi, buram untuk pengguna dan administrator kumpulan pengguna, dan hanya dapat dibaca oleh kumpulan pengguna Anda.

## Mengautentikasi dengan token

Saat pengguna masuk ke aplikasi Anda, Amazon Cognito memverifikasi informasi masuk. Jika login berhasil, Amazon Cognito membuat sesi dan mengembalikan token ID, token akses, dan token penyegaran untuk pengguna yang diautentikasi. Anda dapat menggunakan token untuk memberi pengguna akses ke sumber daya hilir dan API seperti Amazon API Gateway. Atau Anda dapat menukarnya dengan AWS kredensi sementara untuk mengakses lainnya. Layanan AWS



## Menyimpan token

Aplikasi Anda harus dapat menyimpan token dengan berbagai ukuran. Ukuran token dapat berubah karena alasan termasuk, namun tidak terbatas pada, klaim tambahan, perubahan algoritma pengkodean, dan perubahan algoritma enkripsi. Saat Anda mengaktifkan pencabutan token di kumpulan pengguna, Amazon Cognito menambahkan klaim tambahan ke Token Web JSON, meningkatkan ukurannya. Klaim baru `origin_jti` dan `jti` ditambahkan ke token akses dan ID. Untuk informasi selengkapnya tentang pencabutan token, lihat [Mencabut token](#).

### **⚠ Important**

Sebagai praktik terbaik, amankan semua token dalam perjalanan dan penyimpanan dalam konteks aplikasi Anda. Token dapat berisi informasi identifikasi pribadi tentang pengguna Anda, dan informasi tentang model keamanan yang Anda gunakan untuk kumpulan pengguna Anda.

## Menyesuaikan token

Anda dapat menyesuaikan akses dan token ID yang diteruskan Amazon Cognito ke aplikasi Anda. Dalam [aPemicu Lambda generasi pra token](#), Anda dapat menambahkan, memodifikasi,

dan menekan klaim token. Pemicu pembuatan token pra adalah fungsi Lambda tempat Amazon Cognito mengirimkan serangkaian klaim default. Klaim tersebut mencakup cakupan OAuth 2.0, keanggotaan grup kumpulan pengguna, atribut pengguna, dan lainnya. Fungsi tersebut kemudian dapat mengambil kesempatan untuk membuat perubahan saat runtime dan mengembalikan klaim token yang diperbarui ke Amazon Cognito.

Biaya tambahan berlaku untuk mengakses kustomisasi token dengan acara versi 2. Untuk informasi selengkapnya, lihat [Harga Amazon Cognito](#).

## Topik

- [Menggunakan token ID](#)
- [Menggunakan token akses](#)
- [Menggunakan token penyegaran](#)
- [Mencabut token](#)
- [Memverifikasi JSON Web Token](#)
- [Token cache](#)

## Menggunakan token ID

Token ID adalah [JSON Web Token \(JWT\)](#) yang berisi klaim tentang identitas pengguna yang diautentikasi, seperti `name`, dan `email` `phone_number` Anda dapat menggunakan informasi identitas ini dalam aplikasi Anda. Token ID juga dapat digunakan untuk mengautentikasi pengguna ke server sumber daya atau aplikasi server Anda. Anda juga dapat menggunakan Token ID di luar aplikasi dengan operasi API web Anda. Dalam kasus tersebut, Anda harus memverifikasi tanda tangan token ID sebelum Anda dapat mempercayai klaim apa pun di dalam token ID. Lihat [Memverifikasi JSON Web Token](#).

Anda dapat mengatur token ID kedaluwarsa ke nilai apa pun antara 5 menit dan 1 hari. Anda dapat menetapkan nilai ini per klien aplikasi.

### Important

Saat pengguna Anda masuk dengan UI yang dihosting atau penyedia identitas gabungan (iDP), Amazon Cognito menetapkan cookie sesi yang berlaku selama 1 jam. Jika Anda menggunakan UI atau federasi yang dihosting, dan menentukan durasi minimum kurang dari 1 jam untuk akses dan token ID Anda, pengguna Anda akan tetap memiliki sesi yang valid

hingga cookie kedaluwarsa. Jika pengguna memiliki token yang kedaluwarsa selama sesi satu jam, pengguna dapat menyegarkan token mereka tanpa perlu mengautentikasi ulang.

## Header Token ID

Header berisi dua lembar informasi: ID kunci (`kid`), dan algoritme (`alg`).

```
{
  "kid" : "1234example=",
  "alg" : "RS256"
}
```

### **kid**

ID kunci. Nilainya menunjukkan kunci yang digunakan untuk mengamankan JSON Web Signature (JWS) token. Anda dapat melihat ID kunci penandatanganan kumpulan pengguna di `jwtks_uri` titik akhir.

Untuk informasi selengkapnya tentang `kid` parameter, lihat [parameter header Key identifier \(kid\)](#).

### **alg**

Algoritma kriptografi yang digunakan Amazon Cognito untuk mengamankan token akses. Kolam pengguna menggunakan algoritme kriptografi RS256, yang merupakan tanda tangan RSA dengan SHA-256.

Untuk informasi selengkapnya tentang `alg` parameter, lihat [Parameter header Algorithm \(alg\)](#).

## Muatan default token ID

Ini adalah contoh payload dari token ID. Ini berisi klaim tentang pengguna yang diautentikasi. [Untuk informasi selengkapnya tentang klaim standar OpenID Connect \(OIDC\), lihat daftar klaim standar OIDC.](#) Anda dapat menambahkan klaim desain Anda sendiri dengan [aPemicu Lambda generasi pra token.](#)

```
<header>.{
  "sub": "aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee",
  "cognito:groups": [
```



```
    "test-group-a",
    "test-group-b",
    "test-group-c"
  ],
  "email_verified": true,
  "cognito:preferred_role": "arn:aws:iam::111122223333:role/my-test-role",
  "iss": "https://cognito-idp.us-west-2.amazonaws.com/us-west-2_example",
  "cognito:username": "my-test-user",
  "middle_name": "Jane",
  "nonce": "abcdefg",
  "origin_jti": "aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee",
  "cognito:roles": [
    "arn:aws:iam::111122223333:role/my-test-role"
  ],
  "aud": "xxxxxxxxxxxxexample",
  "identities": [
    {
      "userId": "amzn1.account.EXAMPLE",
      "providerName": "LoginWithAmazon",
      "providerType": "LoginWithAmazon",
      "issuer": null,
      "primary": "true",
      "dateCreated": "1642699117273"
    }
  ],
  "event_id": "64f513be-32db-42b0-b78e-b02127b4f463",
  "token_use": "id",
  "auth_time": 1676312777,
  "exp": 1676316377,
  "iat": 1676312777,
  "jti": "aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee",
  "email": "my-test-user@example.com"
}
.<token signature>
```

## sub

Pengenal unik (UUID), atau subjek, untuk pengguna yang diautentikasi. Nama pengguna mungkin tidak unik di kumpulan pengguna Anda. subKlaim adalah cara terbaik untuk mengidentifikasi pengguna tertentu.

## **cognito:groups**

Array nama grup kumpulan pengguna yang memiliki pengguna Anda sebagai anggota. Grup dapat berupa pengenalan yang ditampilkan ke aplikasi, atau mereka dapat membuat permintaan untuk peran IAM pilihan dari kumpulan identitas.

## **cognito:preferred\_role**

ARN dari peran IAM yang Anda kaitkan dengan grup kumpulan pengguna prioritas tertinggi pengguna Anda. Untuk informasi selengkapnya tentang cara kumpulan pengguna memilih klaim peran ini, lihat [Menetapkan nilai prioritas ke grup](#).

## **iss**

Penyedia identitas yang mengeluarkan token. Klaim memiliki format berikut.

```
https://cognito-idp.<Region>.amazonaws.com/<your user pool ID>
```

## **cognito:username**

Nama pengguna pengguna Anda di kolam pengguna Anda.

## **nonce**

nonce Klaim berasal dari parameter dengan nama yang sama yang dapat Anda tambahkan ke permintaan ke titik akhir OAuth 2.0 `authorize` Anda. Saat Anda menambahkan parameter, nonce klaim disertakan dalam token ID yang dikeluarkan Amazon Cognito, dan Anda dapat menggunakannya untuk mencegah serangan replay. Jika Anda tidak memberikan nonce nilai dalam permintaan Anda, Amazon Cognito secara otomatis membuat dan memvalidasi nonce saat Anda mengautentikasi melalui penyedia identitas pihak ketiga, lalu menambahkannya sebagai nonce klaim ke token ID. Implementasi nonce klaim di Amazon Cognito didasarkan pada standar [OIDC](#).

## **origin\_jti**

Pengidentifikasi pencabutan token yang terkait dengan token penyegaran pengguna Anda. Amazon Cognito mereferensikan `origin_jti` klaim saat memeriksa apakah Anda mencabut token pengguna dengan [Cabut titik akhir](#) atau operasi API. [RevokeToken](#) Saat Anda mencabut token, Amazon Cognito membatalkan semua token akses dan ID dengan nilai yang sama.

`origin_jti`

## **cognito:roles**

Array nama peran IAM yang terkait dengan grup pengguna Anda. Setiap grup kumpulan pengguna dapat memiliki satu peran IAM yang terkait dengannya. Array ini mewakili semua

peran IAM untuk grup pengguna Anda, terlepas dari prioritas. Untuk informasi selengkapnya, lihat [Menambahkan grup ke kumpulan pengguna](#).

## **aud**

Klien aplikasi kumpulan pengguna yang mengautentikasi pengguna Anda. Amazon Cognito memberikan nilai yang sama dalam klaim token akses. `client_id`

## **identities**

Isi `identities` atribut pengguna. Atribut berisi informasi tentang setiap profil penyedia identitas pihak ketiga yang telah Anda tautkan ke pengguna, baik dengan login federasi atau dengan [menautkan pengguna federasi ke](#) profil lokal. Informasi ini berisi nama penyedia mereka, ID unik penyedia mereka, dan metadata lainnya.

## **token\_use**

Tujuan token yang dimaksudkan. Dalam token ID, nilainya adalah `id`.

## **auth\_time**

Waktu otentikasi, dalam format waktu Unix, bahwa pengguna Anda menyelesaikan otentikasi.

## **exp**

Waktu kedaluwarsa, dalam format waktu Unix, token pengguna Anda kedaluwarsa.

## **iat**

Waktu yang dikeluarkan, dalam format waktu Unix, Amazon Cognito mengeluarkan token pengguna Anda.

## **jti**

Pengidentifikasi unik JWT.

Token ID dapat berisi klaim standar OIDC yang didefinisikan dalam klaim standar [OIDC](#). Token ID juga dapat berisi atribut khusus yang Anda tentukan di kumpulan pengguna Anda. Amazon Cognito menulis nilai atribut khusus ke token ID sebagai string terlepas dari jenis atribut.

### Note

Atribut kustom kumpulan pengguna selalu diawali dengan `custom:`.

## Tanda Tangan Token ID

Tanda tangan dari token ID dihitung berdasarkan header dan muatan dari token JWT. Sebelum Anda menerima klaim dalam token ID apa pun yang diterima aplikasi Anda, verifikasi tanda tangan token. Untuk informasi selengkapnya, lihat Memverifikasi Token Web JSON. [Memverifikasi JSON Web Token](#).

## Menggunakan token akses

Token akses kolam pengguna berisi klaim tentang pengguna yang diautentikasi, daftar kelompok pengguna, dan daftar cakupan. Tujuan token akses adalah untuk mengotorisasi operasi API. Kumpulan pengguna Anda menerima token akses untuk mengotorisasi operasi layanan mandiri pengguna. Misalnya, Anda dapat menggunakan token akses untuk memberikan akses pengguna Anda untuk menambah, mengubah, atau menghapus atribut pengguna.

Dengan [cakupan OAuth 2.0](#) dalam token akses, yang berasal dari cakupan khusus yang Anda tambahkan ke kumpulan pengguna, Anda dapat mengizinkan pengguna untuk mengambil informasi dari API. Misalnya, Amazon API Gateway mendukung otorisasi dengan token akses Amazon Cognito. Anda dapat mengisi otorisasi REST API dengan informasi dari kumpulan pengguna Anda, atau menggunakan Amazon Cognito sebagai otorisasi Token Web JSON (JWT) untuk API HTTP. Untuk menghasilkan token akses dengan cakupan khusus, Anda harus memintanya melalui titik [akhir publik](#) kumpulan pengguna Anda.

Token akses pengguna Anda adalah izin untuk meminta informasi lebih lanjut tentang atribut pengguna Anda dari [Titik akhir UserInfo](#). Token akses pengguna Anda juga merupakan izin untuk membaca dan menulis atribut pengguna. Tingkat akses ke atribut yang diberikan token akses Anda bergantung pada izin yang Anda tetapkan ke klien aplikasi, dan cakupan yang Anda berikan dalam token.

Token akses adalah [JSON Web Token \(JWT\)](#). Header untuk token akses memiliki struktur yang sama seperti token ID. Amazon Cognito menandatangani token akses dengan kunci berbeda dari kunci yang menandatangani token ID. Nilai klaim ID kunci akses (kid) tidak akan cocok dengan nilai kid klaim dalam token ID dari sesi pengguna yang sama. Dalam kode aplikasi Anda, verifikasi token ID dan akses token secara independen. Jangan percaya klaim dalam token akses sampai Anda memverifikasi tanda tangan. Untuk informasi selengkapnya, lihat [Memverifikasi JSON Web Token](#). Anda dapat mengatur kedaluwarsa token akses ke nilai apa pun antara 5 menit dan 1 hari. Anda dapat menetapkan nilai ini per klien aplikasi.

**⚠ Important**

Untuk token akses dan ID, jangan tentukan minimal kurang dari satu jam jika Anda menggunakan UI yang dihosting. Amazon Cognito HostedUI menggunakan cookie yang berlaku selama satu jam. Jika Anda memasukkan minimum kurang dari satu jam, Anda tidak akan mendapatkan waktu kedaluwarsa yang lebih rendah.

## Akses header token

Header berisi dua lembar informasi: ID kunci (`kid`), dan algoritme (`alg`).

```
{
  "kid" : "1234example="
  "alg" : "RS256",
}
```

### **kid**

ID kunci. Nilainya menunjukkan kunci yang digunakan untuk mengamankan JSON Web Signature (JWS) token. Anda dapat melihat ID kunci penandatanganan kumpulan pengguna di `jwtks_uri` titik akhir.

Untuk informasi selengkapnya tentang `kid` parameter, lihat [parameter header Key identifier \(kid\)](#).

### **alg**

Algoritma kriptografi yang digunakan Amazon Cognito untuk mengamankan token akses. Kolam pengguna menggunakan algoritme kriptografi RS256, yang merupakan tanda tangan RSA dengan SHA-256.

Untuk informasi selengkapnya tentang `alg` parameter, lihat [Parameter header Algorithm \(alg\)](#).

## Akses muatan default token

Ini adalah muatan sampel dari token akses. Untuk informasi lebih lanjut, lihat klaim [JWT](#). Anda dapat menambahkan klaim desain Anda sendiri dengan a [Pemicu Lambda generasi pra token](#).

```
<header>.
{
```

```

"sub": "aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee",
"device_key": "aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee",
"cognito:groups": [
  "testgroup"
],
"iss": "https://cognito-idp.us-west-2.amazonaws.com/us-west-2_example",
"version": 2,
"client_id": "xxxxxxxxxxxxexample",
"origin_jti": "aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee",
"event_id": "aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee",
"token_use": "access",
"scope": "phone openid profile resourceserver.1/appclient2 email",
"auth_time": 1676313851,
"exp": 1676317451,
"iat": 1676313851,
"jti": "aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee",
"username": "my-test-user"
}
.<token signature>

```

## sub

Pengenal unik (UUID), atau subjek, untuk pengguna yang diautentikasi. Nama pengguna mungkin tidak unik di kumpulan pengguna Anda. subKlaim adalah cara terbaik untuk mengidentifikasi pengguna tertentu.

## cognito:groups

Array nama grup kumpulan pengguna yang memiliki pengguna Anda sebagai anggota.

## iss

Penyedia identitas yang mengeluarkan token. Klaim memiliki format berikut.

`https://cognito-idp.<Region>.amazonaws.com/<your user pool ID>`

## client\_id

Klien aplikasi kumpulan pengguna yang mengautentikasi pengguna Anda. Amazon Cognito memberikan nilai yang sama dalam klaim token ID. aud

## origin\_jti

Pengidentifikasi pencabutan token yang terkait dengan token penyegaran pengguna Anda. Amazon Cognito mereferensikan origin\_jti klaim saat memeriksa apakah Anda mencabut

token pengguna dengan [Cabut titik akhir](#) atau operasi API. [RevokeToken](#) Saat Anda mencabut token, Amazon Cognito membatalkan semua token akses dan ID dengan nilai yang sama.

`origin_jti`

### **token\_use**

Tujuan token yang dimaksudkan. Dalam token akses, nilainya adalah `access`.

### **scope**

Daftar cakupan OAuth 2.0 yang menentukan akses apa yang disediakan token. Token dari [Titik akhir token](#) dapat berisi cakupan apa pun yang didukung klien aplikasi Anda. Token dari login Amazon Cognito API hanya berisi cakupan `aws.cognito.signin.user.admin`

### **auth\_time**

Waktu otentikasi, dalam format waktu Unix, bahwa pengguna Anda menyelesaikan otentikasi.

### **exp**

Waktu kedaluwarsa, dalam format waktu Unix, token pengguna Anda kedaluwarsa.

### **iat**

Waktu yang dikeluarkan, dalam format waktu Unix, Amazon Cognito mengeluarkan token pengguna Anda.

### **jti**

Pengidentifikasi unik JWT.

### **username**

Nama pengguna pengguna Anda di kolam pengguna Anda.

## Tanda tangan token akses

Tanda tangan dari token akses dihitung berdasarkan header dan muatan dari token JWT. Ketika digunakan di luar aplikasi di API web Anda, Anda harus selalu memverifikasi tanda tangan ini sebelum menerima token. Untuk informasi selengkapnya, lihat [Memverifikasi JSON Web Token](#).

## Menggunakan token penyegaran

Anda dapat menggunakan token refresh untuk mengambil ID baru dan token akses. Secara default, token refresh kedaluwarsa 30 hari setelah pengguna aplikasi Anda masuk ke kolam pengguna Anda.

Ketika Anda membuat aplikasi untuk kolam pengguna Anda, Anda dapat mengatur aplikasi refresh token kedaluwarsa untuk setiap nilai antara 60 menit dan 10 tahun.

Mobile SDK for iOS, Mobile SDK for Android, Amplify for iOS, Android, dan Flutter secara otomatis me-refresh ID dan token akses jika token refresh valid (belum kedaluwarsa) hadir. ID dan token akses memiliki validitas minimum yang tersisa 2 menit. Jika token refresh kedaluwarsa, pengguna aplikasi Anda harus melakukan autentikasi ulang dengan masuk lagi ke kolam pengguna Anda. Jika minimum untuk token akses dan token ID diatur ke 5 menit, dan Anda menggunakan SDK, token penyegaran akan terus digunakan untuk mengambil akses baru dan token ID. Anda akan melihat perilaku yang diharapkan dengan minimal 7 menit, bukan 5 menit.

Akun pengguna Anda sendiri tidak kedaluwarsa, selama pengguna telah login setidaknya sekali sebelum batas `UnusedAccountValidityDays` waktu untuk akun baru.

## Mendapatkan akses baru dan token identitas dengan token penyegaran

Gunakan API atau UI yang dihosting untuk memulai otentikasi token penyegaran.

Untuk menggunakan token penyegaran untuk mendapatkan ID baru dan token akses dengan API kumpulan pengguna, gunakan operasi [AdminInitiateAuth](#) atau [InitiateAuth](#) API. Lewatkan `REFRESH_TOKEN_AUTH` untuk parameter `AuthFlow`. Di `AuthParameters` properti `AuthFlow`, berikan token penyegaran pengguna Anda sebagai nilai `"REFRESH_TOKEN"`. Amazon Cognito mengembalikan ID baru dan token akses setelah permintaan API Anda melewati semua tantangan.

### Note

Untuk menggunakan API kumpulan pengguna Amazon Cognito untuk menyegarkan token bagi pengguna UI yang dihosting, buat permintaan. `InitiateAuth`

Anda juga dapat mengirimkan token penyegaran ke [Titik akhir token](#) dalam kumpulan pengguna tempat Anda telah mengonfigurasi domain. Di badan permintaan, sertakan `grant_type` nilai `refresh_token` dan `refresh_token` nilai token penyegaran pengguna Anda.

## Mencabut token penyegaran

Anda dapat mencabut `tokenrefresh` milik pengguna. Untuk informasi selengkapnya tentang pencabutan token, lihat. [Mencabut token](#)



**Note**

Mencabut token penyegaran akan mencabut semua ID dan token akses yang dikeluarkan Amazon Cognito dari permintaan penyegaran dengan token itu.

Pengguna dapat keluar dari semua perangkat tempat mereka saat ini masuk saat Anda mencabut semua token pengguna menggunakan operasi `GlobalSignOut` dan `AdminUserGlobalSignOut` API. Setelah pengguna keluar, efek berikut terjadi.

- Token penyegaran pengguna tidak bisa mendapatkan token baru untuk pengguna.
- Token akses pengguna tidak dapat membuat permintaan API yang diotorisasi token.
- Pengguna harus mengautentikasi ulang untuk mendapatkan token baru. Karena cookie sesi UI yang dihosting tidak kedaluwarsa secara otomatis, pengguna Anda dapat mengautentikasi ulang dengan cookie sesi, tanpa permintaan tambahan untuk kredensial. Setelah Anda mengeluarkan pengguna UI yang dihosting, arahkan mereka ke [Titik akhir logout](#), tempat Amazon Cognito akan menghapus cookie sesi mereka.

Dengan token penyegaran, Anda dapat mempertahankan sesi pengguna di aplikasi Anda untuk waktu yang lama. Seiring waktu, pengguna Anda mungkin ingin membatalkan otorisasi beberapa perangkat tempat mereka masuk, terus menyegarkan sesi mereka. Untuk mengeluarkan pengguna dari satu perangkat, cabut token penyegarannya. Saat pengguna Anda ingin keluar dari semua sesi yang diautentikasi, buat permintaan [GlobalSignOut](#) API. Aplikasi Anda dapat memberi pengguna pilihan seperti Keluar dari semua perangkat. `GlobalSignOut` menerima token akses pengguna yang valid—tidak berubah, belum kedaluwarsa, tidak dicabut—. Karena API ini diberi otorisasi token, satu pengguna tidak dapat menggunakannya untuk memulai keluar untuk pengguna lain.

Namun, Anda dapat membuat permintaan [AdminUserGlobalSignOut](#) API yang Anda otorisasi dengan AWS kredensialnya untuk mengeluarkan pengguna mana pun dari semua perangkat mereka. Aplikasi administrator harus memanggil operasi API ini dengan kredensi AWS pengembang dan meneruskan ID kumpulan pengguna dan nama pengguna pengguna sebagai parameter. API `AdminUserGlobalSignOut` dapat mengeluarkan setiap pengguna di kolam pengguna.

Untuk informasi selengkapnya tentang permintaan yang dapat Anda otorisasi dengan AWS kredensi atau token akses pengguna, lihat [Kumpulan pengguna Amazon Cognito operasi API yang diautentikasi dan tidak diautentikasi](#)

## Mencabut token

Anda dapat mencabut token penyegaran untuk pengguna yang menggunakan API. AWS Ketika Anda mencabut token refresh, semua token akses yang sebelumnya dikeluarkan oleh token refresh menjadi tidak valid. Token refresh lainnya dikeluarkan untuk pengguna tidak terpengaruh.

### Note

[Token JWT](#) mandiri dengan tanda tangan dan waktu kedaluwarsa yang ditetapkan saat token dibuat. Token yang dicabut tidak dapat digunakan dengan panggilan API Amazon Cognito apa pun yang memerlukan token. Namun, token yang dicabut akan tetap valid jika diverifikasi menggunakan pustaka JWT apa pun yang memverifikasi tanda tangan dan kedaluwarsa token.

Anda dapat mencabut token penyegaran untuk klien kumpulan pengguna dengan pencabutan token diaktifkan. Saat Anda membuat klien kolam pengguna baru, token pencabutan diaktifkan secara default.

### Aktifkan pencabutan token

Sebelum Anda dapat mencabut token untuk klien kolam pengguna yang ada, Anda harus mengaktifkan pencabutan token. Anda dapat mengaktifkan pencabutan token untuk klien kumpulan pengguna yang ada menggunakan AWS CLI atau API. AWS Untuk melakukannya, panggil perintah `aws cognito-idp describe-user-pool-client` CLI atau operasi `DescribeUserPoolClient` API untuk mengambil pengaturan saat ini dari klien aplikasi Anda. Kemudian panggil perintah `aws cognito-idp update-user-pool-client` CLI atau operasi `UpdateUserPoolClient` API. Sertakan setelan saat ini dari klien aplikasi Anda dan setel `EnableTokenRevocation` parameternya `true`.

Saat Anda membuat klien kumpulan pengguna baru menggunakan AWS Management Console, the AWS CLI, atau AWS API, pencabutan token diaktifkan secara default.

Setelah Anda mengaktifkan pencabutan token, klaim baru ditambahkan di Amazon Cognito JSON Web Tokens. Klaim `origin_jti` dan `jti` ditambahkan ke token akses dan ID. Klaim ini meningkatkan ukuran akses klien aplikasi dan token ID.

Untuk membuat atau memodifikasi klien aplikasi dengan pencabutan token diaktifkan, sertakan parameter berikut dalam permintaan Anda [CreateUserPoolClient](#) atau [UpdateUserPoolClient](#) API.

```
"EnableTokenRevocation": true
```

## Cabut token

Anda dapat mencabut token penyegaran menggunakan permintaan [RevokeToken](#) API, misalnya dengan perintah CLI [aws cognito-idp revoke-token](#). Anda juga dapat mencabut token menggunakan [Cabut titik akhir](#). Titik akhir ini adalah tersedia setelah Anda menambahkan domain ke kolam pengguna Anda. Anda dapat menggunakan titik akhir pencabutan pada domain yang dihosting Amazon Cognito atau domain kustom Anda sendiri.

### Note

Permintaan Anda untuk mencabut token penyegaran harus menyertakan ID klien yang digunakan untuk mendapatkan token.

Berikut ini adalah isi dari permintaan RevokeToken API contoh.

```
{  
  "ClientId": "1example23456789",  
  "ClientSecret": "abcdef123456789ghijklexample",  
  "Token": "eyJjdHkiOiJKV1QiEXAMPLE"  
}
```

Berikut ini adalah contoh permintaan cURL ke `/oauth2/revoke` titik akhir kumpulan pengguna dengan domain kustom.

```
curl --location 'auth.mydomain.com/oauth2/revoke' \  
--header 'Content-Type: application/x-www-form-urlencoded' \  
--header 'Authorization: Basic Base64Encode(client_id:client_secret)' \  
--data-urlencode 'token=abcdef123456789ghijklexample' \  
--data-urlencode 'client_id=1example23456789'
```

RevokeToken Operasi dan `/oauth2/revoke` titik akhir tidak memerlukan otorisasi tambahan kecuali klien aplikasi Anda memiliki rahasia klien.

## Memverifikasi JSON Web Token

Langkah-langkah ini menjelaskan memverifikasi kolam pengguna JSON Web Token (JWT).

## Topik

- [Prasyarat](#)
- [Validasi token dengan aws-jwt-verify](#)
- [Memahami dan memeriksa token](#)

## Prasyarat

Pustaka, SDK, atau kerangka kerja perangkat lunak Anda mungkin sudah menangani tugas di bagian ini. AWS SDK menyediakan alat untuk penanganan dan pengelolaan token kumpulan pengguna Amazon Cognito di aplikasi Anda. AWS Amplify termasuk fungsi untuk mengambil dan menyegarkan token Amazon Cognito.

Untuk informasi lebih lanjut, lihat halaman berikut.

- [Mengintegrasikan otentikasi dan otorisasi Amazon Cognito dengan aplikasi web dan seluler](#)
- [Contoh kode untuk Penyedia Identitas Amazon Cognito menggunakan SDK AWS](#)
- [Alur kerja lanjutan](#) di Amplify Dev Center

Banyak perpustakaan tersedia untuk decoding dan verifikasi JSON Web Token (JWT). Jika Anda ingin memproses token secara manual untuk pemrosesan API sisi server, atau jika Anda menggunakan bahasa pemrograman lain, pustaka ini dapat membantu. Lihat [daftar pustaka dasar OpenID untuk bekerja dengan](#) token JWT.

## Validasi token dengan aws-jwt-verify

Di aplikasi Node.js, AWS merekomendasikan [aws-jwt-verifylibrary](#) untuk memvalidasi parameter dalam token yang diteruskan pengguna ke aplikasi Anda. Dengan `aws-jwt-verify`, Anda dapat mengisi `CognitoJwtVerifier` dengan nilai klaim yang ingin Anda verifikasi untuk satu atau beberapa kumpulan pengguna. Beberapa nilai yang dapat diperiksa termasuk yang berikut ini.

- Akses atau token ID itu tidak cacat atau kedaluwarsa, dan memiliki tanda tangan yang valid.
- Token akses itu berasal dari [kumpulan pengguna dan klien aplikasi yang benar](#).
- Klaim token akses itu berisi cakupan [OAuth 2.0 yang benar](#).
- Bahwa kunci yang menandatangani akses dan token ID Anda [cocok dengan kunci penandatanganan kid dari URI JWKS kumpulan pengguna Anda](#).

URI JWKS berisi informasi publik tentang kunci pribadi yang menandatangani token pengguna Anda. Anda dapat menemukan URI JWKS untuk kumpulan pengguna Anda di `https://cognito-idp.<Region>.amazonaws.com/<userPoolId>/well-known/jwks.json`

Untuk informasi selengkapnya dan contoh kode yang dapat Anda gunakan di aplikasi Node.js atau AWS Lambda otorisasi, lihat [aws-jwt-verify](#) di GitHub.

## Memahami dan memeriksa token

Sebelum mengintegrasikan pemeriksaan token dengan aplikasi, pertimbangkan cara Amazon Cognito merakit JWT. Ambil contoh token dari kumpulan pengguna Anda. Mendekode dan memeriksa mereka secara rinci untuk memahami karakteristik mereka, dan menentukan apa yang ingin Anda verifikasi dan kapan. Misalnya, Anda mungkin ingin memeriksa keanggotaan grup dalam satu skenario, dan cakupan di skenario lain.

Bagian berikut menjelaskan proses untuk memeriksa Amazon Cognito JWT secara manual saat Anda menyiapkan aplikasi.

### Konfirmasikan struktur JWT

JSON Web Token (JWT) mencakup tiga bagian dengan pembatas `.` (titik) di antara mereka.

#### Header

ID kunci, `kid`, dan algoritma RSA, `alg`, yang digunakan Amazon Cognito untuk menandatangani token. Amazon Cognito menandatangani token dengan `alg: RS256`

#### Muatan

Klaim token. Dalam token ID, klaim mencakup atribut pengguna dan informasi tentang kumpulan pengguna `iss`, dan klien aplikasi, `aud`. Dalam token akses, payload mencakup cakupan, keanggotaan grup, kumpulan pengguna sebagai `iss`, dan klien aplikasi Anda sebagai `client_id`

#### Tanda tangan

Tanda tangan tidak dapat didekodekan base64 seperti header dan payload. Ini adalah pengidentifikasi RSA256 yang berasal dari kunci penandatanganan dan parameter yang dapat Anda amati di URI JWKS Anda.

Header dan payload adalah JSON yang dikodekan base64. Anda dapat mengidentifikasi mereka dengan karakter pembuka eyJ yang memecahkan kode ke karakter { awal. Jika pengguna Anda menampilkan JWT yang disandikan base64 ke aplikasi Anda dan tidak dalam format [JSON Header].[JSON Payload].[Signature], itu bukan token Amazon Cognito yang valid dan Anda dapat membuangnya.

## Validasi JWT

Tanda tangan JWT adalah kombinasi hash dari header dan muatan. Amazon Cognito menghasilkan dua pasang kunci kriptografi RSA untuk setiap kolam pengguna. Satu kunci pribadi menandatangani token akses, dan yang lainnya menandatangani token ID.

Untuk memverifikasi tanda tangan dari token JWT

1. Decode ID token.

OpenID Foundation juga [mempertahankan daftar perpustakaan untuk bekerja dengan token JWT](#).

Anda juga dapat menggunakan AWS Lambda untuk memecahkan kode JWT kumpulan pengguna. Untuk informasi selengkapnya, lihat [Mendekode dan memverifikasi token Amazon Cognito JWT](#) menggunakan AWS Lambda

2. Bandingkan ID kunci lokal (kid) dengan publikid.
  - a. Unduh dan simpan JSON Web Key (JWK) publik yang sesuai untuk kolam pengguna Anda. Ini tersedia sebagai bagian dari JSON Web Key Set (JWKS). Anda dapat menemukannya dengan membuat jwks\_uri URI berikut untuk lingkungan Anda:

```
https://cognito-idp.<Region>.amazonaws.com/<userPoolId>/well-known/jwks.json
```

Untuk informasi lebih lanjut tentang JWK dan JWK set, lihat [Kunci Web JSON \(JWK\)](#).

### Note

Amazon Cognito mungkin memutar kunci penandatanganan di kumpulan pengguna Anda. Sebagai praktik terbaik, cache kunci publik di aplikasi Anda, gunakan kid sebagai kunci cache, dan segarkan cache secara berkala. Bandingkan kid token yang diterima aplikasi Anda dengan cache Anda.

Jika Anda menerima token dengan penerbit yang benar tetapi berbedakid, Amazon Cognito mungkin telah memutar kunci penandatanganan. Segarkan cache dari `jwtks_uri` titik akhir kumpulan pengguna Anda.

Ini adalah file `jwtks.json` sampel:

```
{
  "keys": [{
    "kid": "1234example=",
    "alg": "RS256",
    "kty": "RSA",
    "e": "AQAB",
    "n": "1234567890",
    "use": "sig"
  }, {
    "kid": "5678example=",
    "alg": "RS256",
    "kty": "RSA",
    "e": "AQAB",
    "n": "987654321",
    "use": "sig"
  }]
}
```

### ID Kunci (**kid**)

`kid` ini adalah petunjuk yang menunjukkan kunci mana yang digunakan untuk mengamankan JSON Web Signature (JWS) dari token.

### Algoritme (**alg**)

Parameter header `alg` mewakili algoritme kriptografi yang digunakan untuk mengamankan token ID. Kolam pengguna menggunakan algoritme kriptografi RS256, yang merupakan tanda tangan RSA dengan SHA-256. Untuk informasi lebih lanjut tentang RSA, lihat kriptografi [RSA](#).

### Tipe kunci (**kty**)

Parameter `kty` mengidentifikasi keluarga algoritme kriptografi yang digunakan dengan kunci, seperti “RSA” dalam contoh ini.

### Eksponen RSA (**e**)

Parameter **e** berisi nilai eksponen untuk kunci publik RSA. Hal ini direpresentasikan sebagai nilai Base64Urluint-dikodekan.

### Modulus RSA (**n**)

Parameter **n** berisi modulus eksponen untuk kunci publik RSA. Hal ini direpresentasikan sebagai nilai Base64Urluint-dikodekan.

### Gunakan (**use**)

Parameter **use** menjelaskan tujuan penggunaan kunci publik. Untuk contoh ini, nilai **use sig** merupakan tanda tangan.

- b. Cari JSON Web Key publik untuk **kid** yang cocok dengan JWT Anda. **kid**
3. Gunakan pustaka JWT untuk membandingkan tanda tangan penerbit dengan tanda tangan di token. Tanda tangan penerbit berasal dari kunci publik (modulus RSA"**n**") dari **kid** in **jwt**.json yang cocok dengan token. **kid** Anda mungkin perlu mengonversi JWK ke format PEM terlebih dahulu. Contoh berikut mengambil JWT dan JWK dan menggunakan perpustakaan Node.js, [jsonwebtoken](#), untuk memverifikasi tanda tangan JWT:

### Node.js

```
var jwt = require('jsonwebtoken');
var jwkToPem = require('jwk-to-pem');
var pem = jwkToPem(jwk);
jwt.verify(token, pem, { algorithms: ['RS256'] }, function(err, decodedToken) {
});
```

### Verifikasi klaim

#### Untuk memverifikasi klaim JWT

1. Dengan salah satu metode berikut, verifikasi bahwa token belum kedaluwarsa.
  - a. Dekode token dan bandingkan **exp** klaim dengan waktu saat ini.
  - b. Jika token akses Anda menyertakan `aws.cognito.signin.user.admin` klaim, kirim permintaan ke API seperti [GetUser](#). Permintaan API yang Anda [otorisasi dengan token akses](#) mengembalikan kesalahan jika token Anda telah kedaluwarsa.

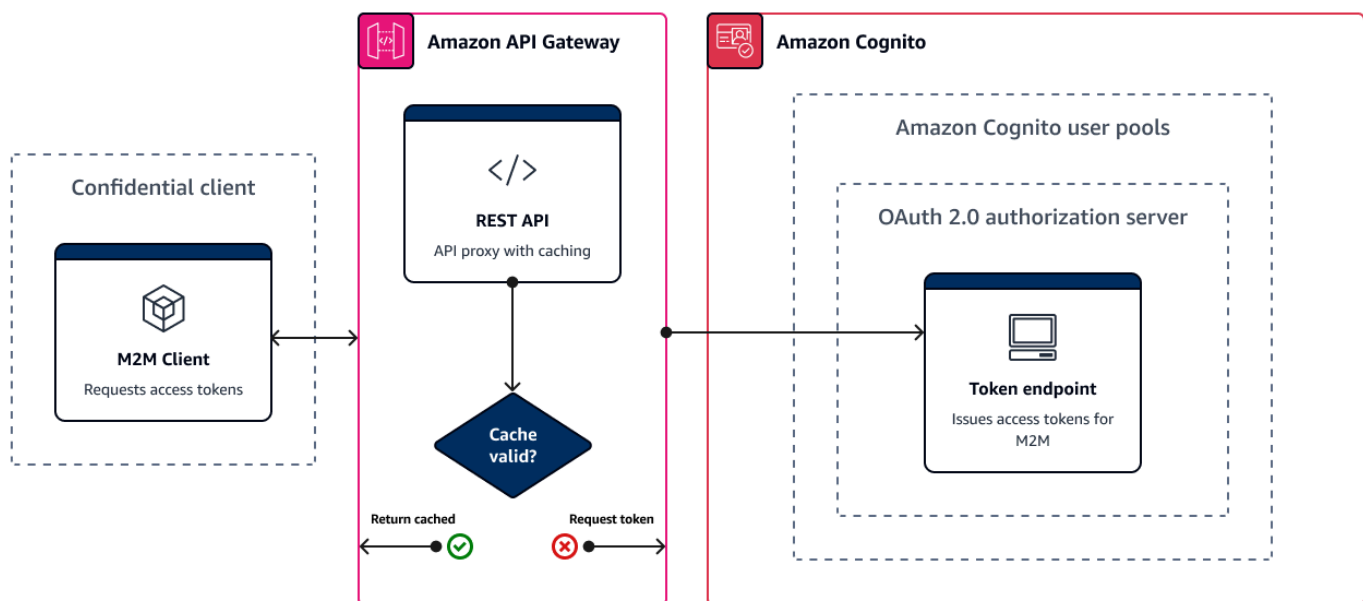


- c. Tunjukkan token akses Anda dalam permintaan ke [Titik akhir UserInfo](#). Permintaan Anda mengembalikan kesalahan jika token Anda telah kedaluwarsa.
2. `aud` klaim dalam token ID dan `client_id` klaim dalam token akses harus cocok dengan ID klien aplikasi yang dibuat di kumpulan pengguna Amazon Cognito.
3. Klaim penerbit (`iss`) harus sesuai dengan kolam pengguna Anda. Sebagai contoh, kolam pengguna yang dibuat di Wilayah `us-east-1` akan memiliki nilai `iss` sebagai berikut:
 

```
https://cognito-idp.us-east-1.amazonaws.com/<userpoolID>.
```
4. Periksa klaim `token_use`.
  - Jika Anda hanya menerima token akses dalam operasi API web Anda, nilainya harus `access`.
  - Jika Anda hanya menggunakan ID token, nilainya harus `id`.
  - Jika Anda menggunakan ID dan token akses, klaim `token_use` harus `id` atau `access`.

Anda sekarang dapat mempercayai klaim di dalam token.

## Token cache



Aplikasi Anda harus berhasil menyelesaikan salah satu permintaan berikut setiap kali Anda ingin mendapatkan JSON Web Token (JWT) baru.

- Minta kredensi klien atau [hibah](#) kode otorisasi dari [Titik akhir token](#)

- Minta hibah implisit dari UI yang dihosting.
- Mengautentikasi pengguna lokal dalam permintaan Amazon Cognito API seperti [InitiateAuth](#)

Anda dapat mengonfigurasi kumpulan pengguna untuk mengatur token agar kedaluwarsa dalam hitungan menit, jam, atau hari. Untuk memastikan performa dan ketersediaan aplikasi Anda, gunakan token Amazon Cognito hingga token tersebut kedaluwarsa, lalu ambil token baru. Solusi cache yang Anda buat untuk aplikasi membuat token tetap tersedia, dan mencegah penolakan permintaan oleh Amazon Cognito jika tingkat permintaan Anda terlalu tinggi. Aplikasi sisi klien harus menyimpan token dalam cache memori. Aplikasi sisi server dapat menambahkan mekanisme cache terenkripsi untuk menyimpan token.

Saat kumpulan pengguna menghasilkan volume pengguna atau machine-to-machine aktivitas yang tinggi, Anda mungkin menemukan batasan yang ditetapkan Amazon Cognito pada jumlah permintaan token yang dapat Anda buat. Untuk mengurangi jumlah permintaan yang Anda buat ke titik akhir Amazon Cognito, Anda dapat menyimpan dan menggunakan kembali data autentikasi dengan aman, atau menerapkan backoff dan percobaan ulang eksponensial.

Data otentikasi berasal dari dua kelas endpoint. Titik akhir Amazon Cognito [OAuth 2.0 menyertakan titik akhir](#) token, yang melayani kredensi klien dan permintaan kode otorisasi UI yang dihosting. [Titik akhir layanan](#) menjawab permintaan API kumpulan pengguna seperti `InitiateAuth` dan `RespondToAuthChallenge`. Setiap jenis permintaan memiliki batasnya sendiri. Untuk informasi selengkapnya tentang batasan, lihat [Kuota di Amazon Cognito](#).

## Caching token machine-to-machine akses dengan Amazon API Gateway


Dengan caching token API Gateway, aplikasi Anda dapat menskalakan sebagai respons terhadap peristiwa yang lebih besar dari kuota tingkat permintaan default titik akhir Amazon Cognito OAuth.

Anda dapat men-cache token akses sehingga aplikasi Anda hanya meminta token akses baru jika token cache kedaluwarsa. Jika tidak, titik akhir caching Anda mengembalikan token dari cache. Ini mencegah panggilan tambahan ke titik akhir Amazon Cognito API. Bila Anda menggunakan Amazon API Gateway sebagai proxy ke [Titik akhir token](#), API Anda merespons sebagian besar permintaan yang sebaliknya akan berkontribusi pada kuota permintaan Anda, menghindari permintaan yang gagal sebagai akibat dari pembatasan tarif.

Solusi berbasis API Gateway berikut menawarkan implementasi caching token dengan latensi rendah, kode rendah/tanpa kode. API Gateway API dienkripsi saat transit, dan secara opsional diam. Cache API Gateway sangat ideal untuk hibah [kredensi klien OAuth 2.0, jenis hibah](#) volume tinggi yang sering menghasilkan token akses untuk sesi otorisasi dan layanan mikro. machine-

to-machine Dalam peristiwa seperti lonjakan lalu lintas yang menyebabkan layanan mikro Anda berskala horizontal, Anda dapat berakhir dengan banyak sistem yang menggunakan kredensi klien yang sama pada volume yang melebihi batas AWS tingkat permintaan kumpulan pengguna atau klien aplikasi Anda. Untuk menjaga ketersediaan aplikasi dan latensi rendah, solusi caching adalah praktik terbaik dalam skenario seperti itu.


Dalam solusi ini, Anda menentukan cache di API Anda untuk menyimpan token akses terpisah untuk setiap kombinasi cakupan OAuth dan klien aplikasi yang ingin Anda minta di aplikasi Anda. Saat aplikasi Anda membuat permintaan yang cocok dengan kunci cache, API Anda merespons dengan token akses yang dikeluarkan Amazon Cognito ke permintaan pertama yang cocok dengan kunci cache. Ketika durasi kunci cache Anda kedaluwarsa, API Anda meneruskan permintaan ke titik akhir token Anda dan menyimpan token akses baru.

 Note

Durasi kunci cache Anda harus lebih pendek dari durasi token akses klien aplikasi Anda.

Kunci cache adalah kombinasi dari cakupan OAuth yang Anda minta dalam parameter scope URL dan Authorization header dalam permintaan. AuthorizationHeader berisi ID klien aplikasi dan rahasia klien Anda. Anda tidak perlu menerapkan logika tambahan di aplikasi Anda untuk mengimplementasikan solusi ini. Anda hanya harus memperbarui konfigurasi untuk mengubah jalur ke titik akhir token kumpulan pengguna Anda.

Anda juga dapat menerapkan token caching dengan [ElastiCache for Redis](#). [Untuk kontrol halus dengan kebijakan AWS Identity and Access Management \(IAM\), pertimbangkan cache Amazon DynamoDB](#).

 Note

Caching di API Gateway dikenakan biaya tambahan. [Lihat harga untuk lebih jelasnya](#).

Untuk menyiapkan proxy caching dengan API Gateway

1. Buka [konsol API Gateway](#) dan buat REST API.
2. Di Sumber Daya, buat metode POST.
  - a. Pilih tipe Integrasi HTTP.

- b. Pilih Gunakan integrasi proxy HTTP.
  - c. Masukkan URL Endpoint dari. `https://<your user pool domain>/oauth2/token`
3. Di Sumber Daya, konfigurasi kunci cache.
  - a. Edit permintaan Metode metode POST Anda.
  - b. Tetapkan scope parameter dan Authorization header Anda sebagai kunci caching Anda.
    - i. Tambahkan string kueri ke parameter string kueri URL dan pilih Caching untuk scope string.
    - ii. Tambahkan header ke header permintaan HTTP dan pilih Caching untuk header. `Authorization`
4. Dalam Tahapan, konfigurasi caching.
  - a. Pilih tahap yang ingin Anda modifikasi.
  - b. Di bawah Pengaturan, pilih Aktifkan cache API.
  - c. Pilih kapasitas Cache.
  - d. Pilih Cache time-to-live (TTL) minimal 3600 detik.
  - e. Kosongkan kotak centang Memerlukan otorisasi.
5. Dalam Tahapan, perhatikan URL Panggilan.
6. Perbarui aplikasi Anda ke permintaan token POST ke URL Invoke API Anda, bukan `/oauth2/token` titik akhir kumpulan pengguna Anda.

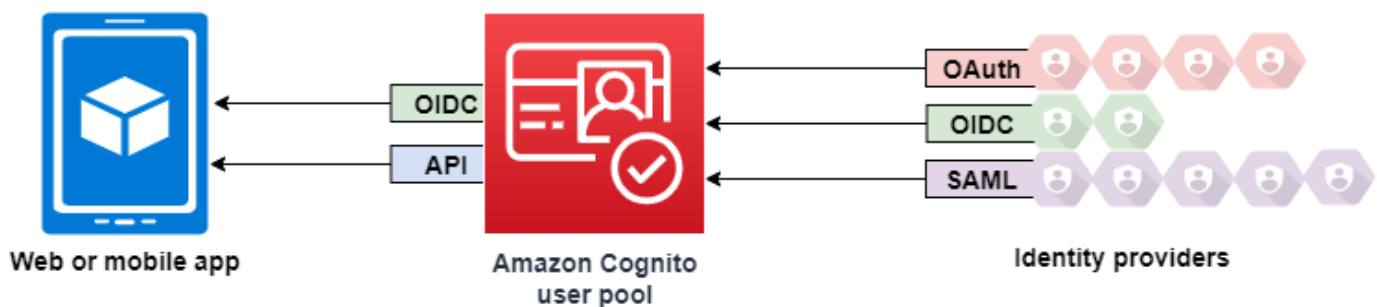
## Mengakses sumber daya setelah otentikasi kumpulan pengguna berhasil

Pengguna aplikasi Anda dapat masuk langsung melalui kumpulan pengguna, atau mereka dapat melakukan federasi melalui penyedia identitas pihak ketiga (iDP). Kumpulan pengguna mengelola overhead penanganan token yang dikembalikan dari login sosial melalui Facebook, Google, Amazon, dan Apple, dan dari OpenID Connect (OIDC) dan SAMP. IdPs Untuk informasi selengkapnya, lihat [Menggunakan token dengan kolam pengguna](#).

Setelah autentikasi berhasil, aplikasi Anda akan menerima token kolam pengguna dari Amazon Cognito. Anda dapat menggunakan token kumpulan pengguna untuk:

- Ambil AWS kredensial yang mengotorisasi permintaan untuk sumber daya aplikasi seperti Amazon Layanan AWS DynamoDB dan Amazon S3.
- Berikan bukti otentikasi sementara yang dapat dibatalkan.
- Isi data identitas ke profil pengguna di aplikasi Anda.
- Otorisasi perubahan pada profil pengguna yang masuk di direktori kumpulan pengguna.
- Otorisasi permintaan informasi pengguna dengan token akses.
- Otorisasi permintaan ke data yang berada di belakang API eksternal yang dilindungi akses dengan token akses.
- Otorisasi akses ke aset aplikasi yang disimpan di klien atau server dengan Izin Terverifikasi Amazon.

Untuk informasi selengkapnya, lihat [Alur otentikasi kumpulan pengguna](#) dan [Menggunakan token dengan kolam pengguna](#).



## Topik

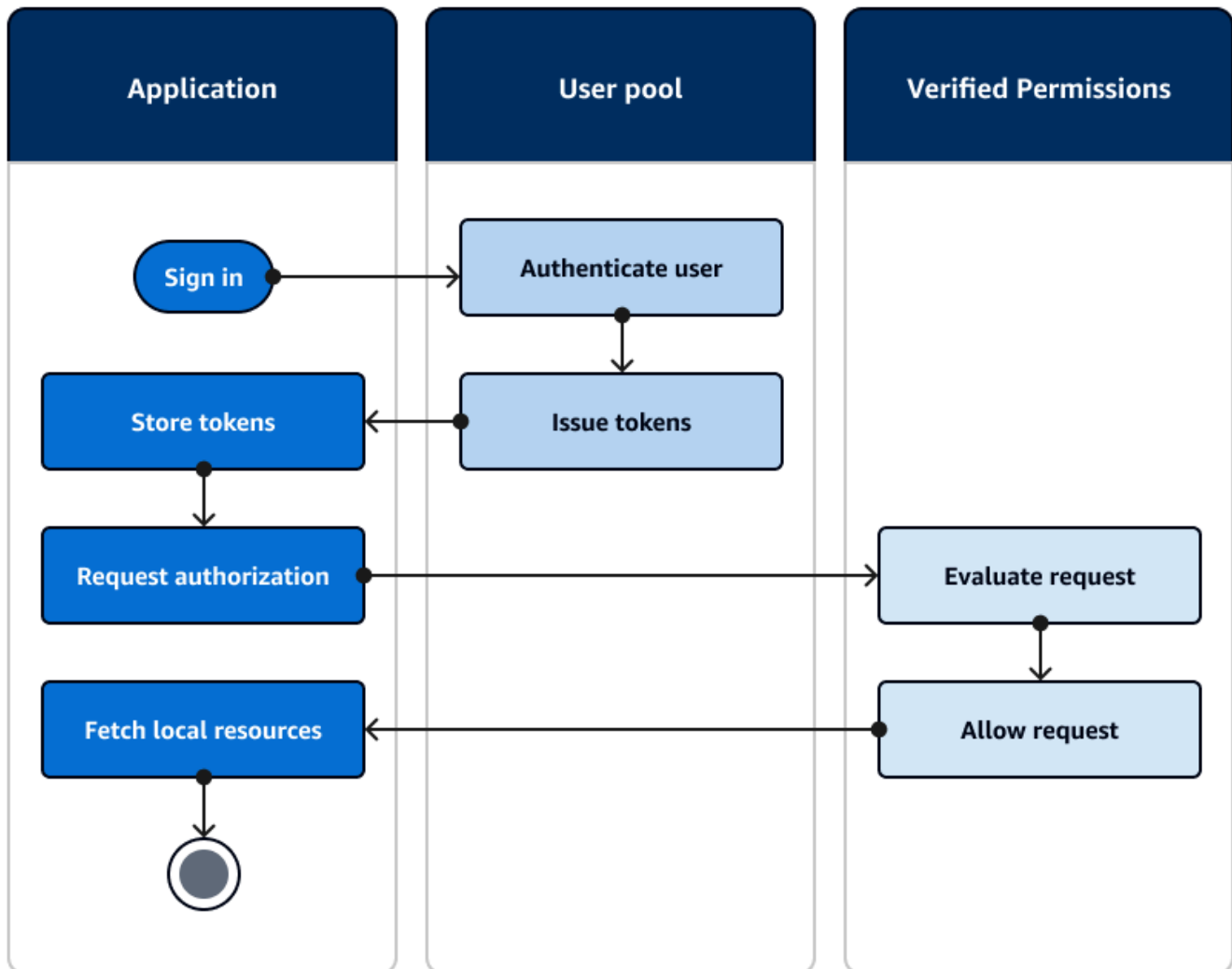
- [Mengotorisasi akses ke sumber daya klien atau server dengan Izin Terverifikasi Amazon](#)
- [Mengakses sumber daya dengan API Gateway setelah login](#)
- [Mengakses Layanan AWS menggunakan kumpulan identitas setelah masuk](#)

## Mengotorisasi akses ke sumber daya klien atau server dengan Izin Terverifikasi Amazon

Aplikasi Anda dapat meneruskan token dari pengguna yang masuk ke Izin Terverifikasi [Amazon](#). Izin Terverifikasi adalah layanan pengelolaan izin dan otorisasi yang dapat diskalakan dan berbutir halus untuk aplikasi kustom yang telah Anda buat. Kumpulan pengguna Amazon Cognito dapat menjadi sumber identitas ke toko kebijakan Izin Terverifikasi. Izin Terverifikasi membuat keputusan otorisasi

untuk tindakan dan sumber daya yang diminta, seperti `GetPhoto untukpremium_badge.png`, dari prinsipal dan atributnya dalam token kumpulan pengguna.

Diagram berikut menunjukkan bagaimana aplikasi Anda dapat meneruskan token pengguna ke Izin Terverifikasi dalam permintaan otorisasi.



## Memulai Izin Terverifikasi Amazon

Setelah mengintegrasikan kumpulan pengguna dengan Izin Terverifikasi, Anda mendapatkan sumber otorisasi terperinci pusat untuk semua aplikasi Amazon Cognito Anda. Ini menghilangkan kebutuhan akan logika keamanan berbutir halus yang seharusnya Anda kodekan dan replikasi di antara semua aplikasi Anda. Untuk informasi selengkapnya tentang otorisasi dengan Izin Terverifikasi, lihat.

[Otorisasi dengan Izin Terverifikasi Amazon](#)

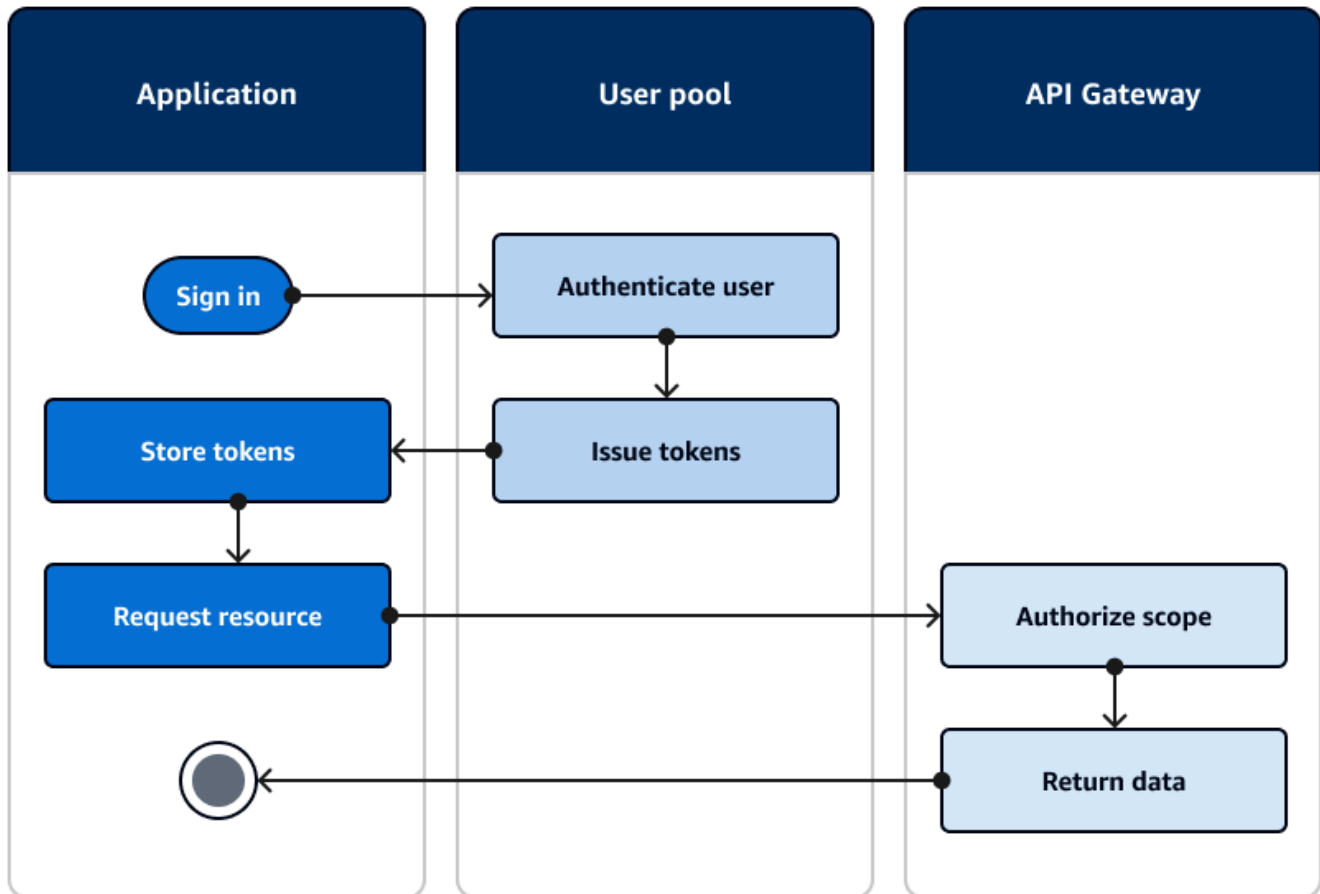
Permintaan otorisasi Izin Terverifikasi memerlukan AWS kredensial. Anda dapat menerapkan beberapa teknik berikut untuk menerapkan kredensial dengan aman ke permintaan otorisasi.

- Mengoperasikan aplikasi web yang dapat menyimpan rahasia di backend server.
- Dapatkan kredensial kumpulan identitas yang diautentikasi.
- Permintaan pengguna proxy melalui access-token-authorized API, dan tambahkan AWS kredensial ke permintaan.

## Mengakses sumber daya dengan API Gateway setelah login

Penggunaan umum token kumpulan pengguna Amazon Cognito adalah untuk mengotorisasi permintaan ke API [REST API Gateway API](#). Cakupan OAuth 2.0 dalam token akses dapat mengotorisasi metode dan jalur, seperti untuk HTTP GET /app\_assets Token ID dapat berfungsi sebagai otentikasi generik ke API dan dapat meneruskan atribut pengguna ke layanan backend. API Gateway memiliki opsi otorisasi khusus tambahan seperti otorisasi [JWT untuk API HTTP dan otorisasi Lambda yang dapat menerapkan logika yang lebih halus](#).

Diagram berikut menggambarkan aplikasi yang mendapatkan akses ke REST API dengan cakupan OAuth 2.0 dalam token akses.



Aplikasi Anda harus mengumpulkan token dari sesi yang diautentikasi dan menambahkannya sebagai token pembawa ke Authorization header dalam permintaan. Konfigurasi otorisasi yang Anda konfigurasi untuk API, jalur, dan metode untuk mengevaluasi konten token. API Gateway mengembalikan data hanya jika permintaan cocok dengan kondisi yang Anda siapkan untuk otorisasi.

Beberapa cara potensial API Gateway API dapat menyetujui akses dari aplikasi adalah:

- Token akses berisi cakupan OAuth 2.0 yang benar. [Otorisasi kumpulan pengguna Amazon Cognito untuk REST API](#) adalah implementasi umum dengan penghalang masuk yang rendah. Anda juga dapat mengevaluasi isi, parameter string kueri, dan header permintaan ke jenis otorisasi ini.
- Token ID valid dan tidak kedaluwarsa. Saat Anda meneruskan token ID ke otorisasi Amazon Cognito, Anda dapat melakukan validasi tambahan dari konten token ID di server aplikasi Anda.
- Grup, klaim, atribut, atau peran dalam token akses atau ID memenuhi persyaratan yang Anda tetapkan dalam fungsi Lambda. [Authorizer Lambda mem-parsing token di header permintaan dan](#)



[mengevaluasinya untuk keputusan otorisasi](#). Anda dapat membuat logika kustom dalam fungsi Anda atau membuat permintaan API ke [Izin Terverifikasi Amazon](#).

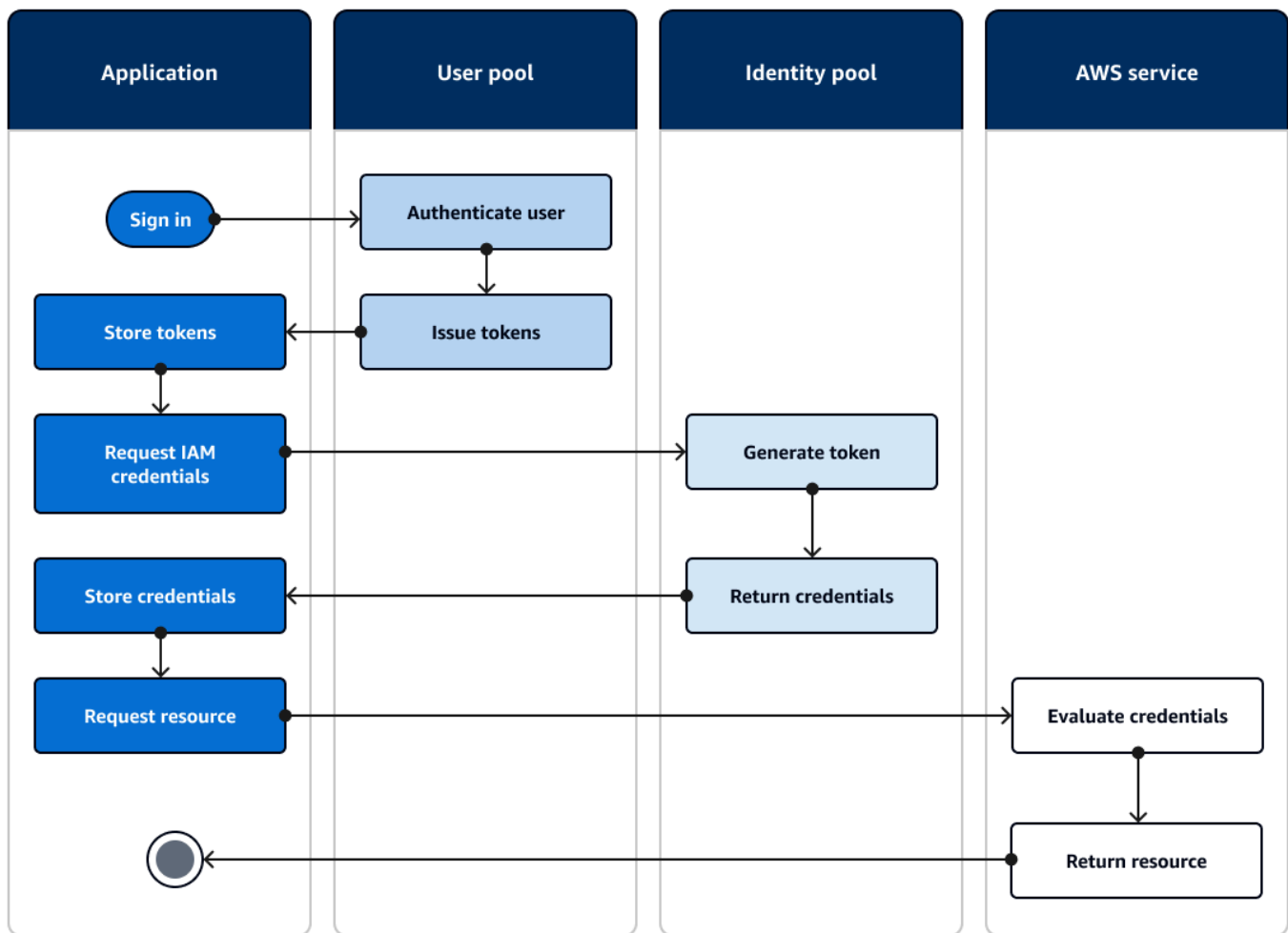
Anda juga dapat mengotorisasi permintaan ke [AWS AppSync GraphQL](#) API dengan token dari kumpulan pengguna.

## Mengakses Layanan AWS menggunakan kumpulan identitas setelah masuk

Setelah pengguna Anda masuk dengan kumpulan pengguna, mereka dapat mengakses Layanan AWS dengan kredensial API sementara yang dikeluarkan dari kumpulan identitas.

Aplikasi web atau seluler Anda menerima token dari kumpulan pengguna. Saat Anda mengonfigurasi kumpulan pengguna sebagai penyedia identitas ke kumpulan identitas Anda, kumpulan identitas akan menukar token untuk AWS kredensial sementara. Kredensial ini dapat dicakup ke peran IAM dan kebijakannya yang memberi pengguna akses ke serangkaian sumber daya terbatas. AWS Untuk informasi selengkapnya, lihat [Aliran otentikasi kumpulan identitas \(identitas federasi\)](#).

Diagram berikut menunjukkan cara aplikasi masuk dengan kumpulan pengguna, mengambil kredensial kumpulan identitas, dan meminta aset dari sebuah Layanan AWS



Anda dapat menggunakan kredensial kumpulan identitas untuk:

- Buat permintaan otorisasi berbutir halus ke Izin Terverifikasi Amazon dengan kredensial pengguna Anda sendiri.
- Connect ke Amazon API Gateway REST API atau AWS AppSync GraphQL API yang mengotorisasi koneksi dengan IAM.
- Connect ke backend database seperti Amazon DynamoDB atau Amazon RDS yang mengotorisasi koneksi dengan IAM.
- Ambil aset aplikasi dari bucket Amazon S3.
- Memulai sesi dengan desktop WorkSpaces virtual Amazon.

Kumpulan identitas tidak beroperasi secara eksklusif dalam sesi yang diautentikasi dengan kumpulan pengguna. Mereka juga menerima otentikasi langsung dari penyedia identitas pihak ketiga dan dapat menghasilkan kredensial untuk pengguna tamu yang tidak diautentikasi.

Untuk informasi selengkapnya tentang penggunaan kumpulan identitas bersama dengan grup kumpulan pengguna untuk mengontrol akses ke AWS sumber daya Anda, lihat [Menambahkan grup ke kumpulan pengguna](#) dan [Menggunakan kontrol akses berbasis peran](#). Juga, untuk informasi lebih lanjut tentang kumpulan identitas dan AWS Identity and Access Management, lihat [Konsep kolom identitas](#).

## Menyiapkan kolom pengguna dengan AWS Management Console

Buat kolom pengguna Amazon Cognito dan buat catatan ID Kolam Pengguna dan ID Klien Aplikasi untuk setiap aplikasi klien Anda. Untuk informasi selengkapnya tentang membuat kolom pengguna ini, lihat [Memulai dengan kumpulan pengguna](#).

## Menyiapkan kolom identitas dengan AWS Management Console

Prosedur berikut menjelaskan cara menggunakan kumpulan identitas AWS Management Console untuk mengintegrasikan kumpulan identitas dengan satu atau beberapa kumpulan pengguna dan aplikasi klien.

Untuk menambahkan penyedia identitas kumpulan pengguna Amazon Cognito (iDP)

1. Pilih kumpulan Identitas dari konsol [Amazon Cognito](#). Pilih kumpulan identitas.
2. Pilih tab Akses pengguna.
3. Pilih Tambahkan penyedia identitas.
4. Pilih kumpulan pengguna Amazon Cognito.
5. Masukkan ID kumpulan Pengguna dan ID klien Aplikasi.
6. Untuk menyetel peran yang diminta Amazon Cognito saat mengeluarkan kredensial kepada pengguna yang telah mengautentikasi dengan penyedia ini, konfigurasi setelan Peran.
  - a. Anda dapat memberi pengguna dari IDP itu peran Default yang Anda atur saat mengonfigurasi peran Terautentikasi, atau Anda dapat Memilih peran dengan aturan. Dengan IdP kumpulan pengguna Amazon Cognito, Anda juga dapat Memilih peran dengan klaim `preferred_role` dalam token. Untuk informasi lebih lanjut tentang `cognito:preferred_role` klaim, lihat [Menetapkan nilai prioritas ke grup](#).

- i. Jika Anda memilih Pilih peran dengan aturan, masukkan Klaim sumber dari autentikasi pengguna Anda, Operator yang ingin Anda gunakan untuk membandingkan klaim dengan aturan, Nilai yang akan menyebabkan kecocokan dengan pilihan peran ini, dan Peran yang ingin Anda tetapkan saat penetapan peran cocok. Pilih Tambahkan yang lain untuk membuat aturan tambahan berdasarkan kondisi yang berbeda.
    - ii. Jika Anda memilih peran Pilih dengan klaim `preferred_role` dalam token, Amazon Cognito akan mengeluarkan kredensi untuk peran tersebut dalam klaim pengguna Anda. `cognito:preferred_role` Jika tidak ada klaim peran pilihan, Amazon Cognito mengeluarkan kredensial berdasarkan resolusi Peran Anda.
  - b. Pilih Resolusi Peran. Jika klaim pengguna tidak sesuai dengan aturan, Anda dapat menolak kredensial atau mengeluarkan kredensi untuk peran Terautentikasi Anda.
7. Untuk mengubah tag utama yang ditetapkan Amazon Cognito saat mengeluarkan kredensial kepada pengguna yang telah diautentikasi dengan penyedia ini, konfigurasi Atribut untuk kontrol akses.
- Untuk tidak menerapkan tag utama, pilih Tidak aktif.
  - Untuk menerapkan tag utama berdasarkan sub dan aud klaim, pilih Gunakan pemetaan default.
  - Untuk membuat skema atribut kustom Anda sendiri ke tag utama, pilih Gunakan pemetaan khusus. Kemudian masukkan kunci Tag yang ingin Anda sumber dari setiap Klaim yang ingin Anda wakili dalam tag.
8. Pilih Simpan perubahan.

## Mengintegrasikan kumpulan pengguna dengan kolam identitas

Setelah pengguna aplikasi Anda diautentikasi, tambahkan token identitas pengguna tersebut ke peta masuk di penyedia kredensialnya. Nama penyedia akan tergantung pada ID kolam pengguna Amazon Cognito Anda. Ini akan memiliki struktur sebagai berikut:

```
cognito-idp.<region>.amazonaws.com/<YOUR_USER_POOL_ID>
```

Anda dapat memperoleh nilai untuk `<region>` dari User Pool ID. Misalnya, jika ID kumpulan pengguna adalah `us-east-1_EXAMPLE1`, maka `<region>` adalah `us-east-1`. Jika ID kumpulan pengguna adalah `us-west-2_EXAMPLE2`, maka `<region>` adalah `us-west-2`.

## JavaScript

```

var cognitoUser = userPool.getCurrentUser();

if (cognitoUser != null) {
  cognitoUser.getSession(function(err, result) {
    if (result) {
      console.log('You are now logged in.');
```

    // Add the User's Id Token to the Cognito credentials login map.

```

      AWS.config.credentials = new AWS.CognitoIdentityCredentials({
        IdentityPoolId: 'YOUR_IDENTITY_POOL_ID',
        Logins: {
          'cognito-idp.<region>.amazonaws.com/<YOUR_USER_POOL_ID>':
result.getIdToken().getJwtToken()
        }
      });
    }
  });
}

```

## Android

```

cognitoUser.getSessionInBackground(new AuthenticationHandler() {
  @Override
  public void onSuccess(CognitoUserSession session) {
    String idToken = session.getIdToken().getJWTToken();

    Map<String, String> logins = new HashMap<String, String>();
    logins.put("cognito-idp.<region>.amazonaws.com/<YOUR_USER_POOL_ID>",
session.getIdToken().getJWTToken());
    credentialsProvider.setLogins(logins);
  }
});

```

## iOS - objective-C

```

AWSServiceConfiguration *serviceConfiguration = [[AWSServiceConfiguration alloc]
initWithRegion:AWSRegionUSEast1 credentialsProvider:nil];
AWSCognitoIdentityUserPoolConfiguration *userPoolConfiguration =
[[AWSCognitoIdentityUserPoolConfiguration alloc] initWithClientId:@"YOUR_CLIENT_ID"
clientSecret:@"YOUR_CLIENT_SECRET" poolId:@"YOUR_USER_POOL_ID"];

```

```
[AWSCognitoIdentityUserPool
  registerCognitoIdentityUserPoolWithConfiguration:serviceConfiguration
  userPoolConfiguration:userPoolConfiguration forKey:@"UserPool"];
AWSCognitoIdentityUserPool *pool = [AWSCognitoIdentityUserPool
  CognitoIdentityUserPoolForKey:@"UserPool"];
AWSCognitoCredentialsProvider *credentialsProvider = [[AWSCognitoCredentialsProvider
  alloc] initWithRegionType:AWSRegionUSEast1 identityPoolId:@"YOUR_IDENTITY_POOL_ID"
  identityProviderManager:pool];
```

## iOS - swift

```
let serviceConfiguration = AWSServiceConfiguration(region: .USEast1,
  credentialsProvider: nil)
let userPoolConfiguration = AWSCognitoIdentityUserPoolConfiguration(clientId:
  "YOUR_CLIENT_ID", clientSecret: "YOUR_CLIENT_SECRET", poolId: "YOUR_USER_POOL_ID")
AWSCognitoIdentityUserPool.registerCognitoIdentityUserPoolWithConfiguration(serviceConfiguration,
  userPoolConfiguration: userPoolConfiguration, forKey: "UserPool")
let pool = AWSCognitoIdentityUserPool(forKey: "UserPool")
let credentialsProvider = AWSCognitoCredentialsProvider(regionType: .USEast1,
  identityPoolId: "YOUR_IDENTITY_POOL_ID", identityProviderManager:pool)
```

## Menggunakan fitur keamanan kumpulan pengguna Amazon Cognito

Anda dapat menambahkan autentikasi Autentikasi Multi-Faktor (MFA) ke kolam pengguna untuk melindungi identitas pengguna Anda. MFA menambahkan faktor otentikasi kedua sehingga kumpulan pengguna Anda tidak hanya mengandalkan nama pengguna dan kata sandi. Anda dapat menggunakan pesan teks SMS atau kata sandi satu kali berbasis waktu (TOTP) sebagai faktor kedua untuk masuk ke pengguna Anda. Anda juga dapat menggunakan autentikasi adaptif dengan model berbasis risiko untuk memprediksi kapan Anda mungkin memerlukan faktor autentikasi lain. Fitur keamanan lanjutan kumpulan pengguna mencakup otentikasi adaptif dan perlindungan terhadap kredensial yang dikompromikan.

### Topik

- [Menambahkan MFA ke kumpulan pengguna](#)
- [Menambahkan keamanan tingkat lanjut ke kumpulan pengguna](#)
- [Mengaitkan ACL AWS WAF web dengan kumpulan pengguna](#)
- [Sensitivitas casing kumpulan pengguna](#)

- [perlindungan penghapusan kolam pengguna](#)
- [Mengelola respons kesalahan keberadaan pengguna](#)

## Menambahkan MFA ke kumpulan pengguna

Autentikasi multi-faktor (MFA) meningkatkan keamanan untuk aplikasi Anda. Ini menambahkan sesuatu yang Anda memiliki faktor otentikasi untuk sesuatu yang Anda tahu faktor nama pengguna dan kata sandi. Anda dapat memilih pesan teks SMS atau kata sandi satu kali berbasis waktu (TOTP) sebagai faktor kedua untuk masuk ke pengguna Anda.

### Note

Pertama kali pengguna baru masuk ke aplikasi Anda, Amazon Cognito mengeluarkan token OAuth 2.0, meskipun kumpulan pengguna Anda memerlukan MFA. Faktor otentikasi kedua saat pengguna Anda masuk untuk pertama kalinya adalah konfirmasi mereka atas pesan verifikasi yang dikirimkan Amazon Cognito kepada mereka. Jika kumpulan pengguna Anda memerlukan MFA, Amazon Cognito meminta pengguna Anda untuk mendaftarkan faktor masuk tambahan untuk digunakan selama setiap upaya masuk setelah yang pertama.

Dengan autentikasi adaptif, Anda dapat mengonfigurasi kolam pengguna Anda untuk memerlukan autentikasi faktor kedua dalam menanggapi tingkat risiko meningkat. Untuk menambahkan autentikasi adaptif ke kolam pengguna Anda, lihat [Menambahkan keamanan tingkat lanjut ke kumpulan pengguna](#).

Saat Anda mengatur `MFA required` untuk kumpulan pengguna, semua pengguna harus menyelesaikan MFA untuk masuk. Untuk masuk, setiap pengguna harus mengatur setidaknya satu faktor MFA, seperti SMS atau TOTP. Ketika Anda mengatur `MFA kerequired`, Anda harus menyertakan pengaturan MFA dalam orientasi pengguna sehingga kumpulan pengguna Anda mengizinkan mereka untuk masuk.

Jika Anda mengaktifkan SMS sebagai faktor MFA, Anda dapat meminta pengguna memberikan nomor telepon dan meminta pengguna Anda memverifikasi mereka saat mendaftar. Jika Anda telah menetapkan MFA ke `required` dan hanya mendukung SMS sebagai faktor, pengguna harus memberikan nomor telepon. Pengguna tanpa nomor telepon memerlukan dukungan Anda untuk menambahkan nomor telepon ke profil mereka sebelum mereka dapat masuk. Anda dapat menggunakan nomor telepon yang belum diverifikasi untuk SMS MFA. Angka-angka ini akan menerima status terverifikasi setelah MFA berhasil.

Jika Anda telah menetapkan MFA agar diperlukan dan Anda mengaktifkan SMS dan TOTP sebagai metode verifikasi yang didukung, Amazon Cognito meminta pengguna baru tanpa nomor telepon untuk mengatur TOTP MFA. Jika Anda telah menyetel MFA agar diperlukan dan satu-satunya metode MFA yang Anda aktifkan adalah TOTP, Amazon Cognito meminta semua pengguna baru untuk mengatur TOTP MFA saat kedua kalinya mereka masuk. Amazon Cognito menghasilkan tantangan untuk menyiapkan TOTP MFA sebagai respons terhadap dan operasi API.

### [InitiateAuthAdminInitiateAuth](#)

UI yang dihosting meminta pengguna untuk mengatur MFA saat Anda mengatur MFA agar diperlukan. Saat Anda menyetel MFA menjadi opsional di kumpulan pengguna, UI yang dihosting tidak meminta pengguna. Untuk bekerja dengan MFA opsional, Anda harus membuat antarmuka di aplikasi yang meminta pengguna untuk memilih apakah mereka ingin menyiapkan MFA, lalu memandu mereka melalui input API untuk memverifikasi faktor masuk tambahan mereka.

Setelah lima upaya gagal untuk menyajikan kode MFA, Amazon Cognito memulai proses penguncian batas waktu eksponensial yang dijelaskan di [Alur otentikasi kumpulan pengguna](#)

### Topik

- [Prasyarat](#)
- [Mengkonfigurasi otentikasi multi-faktor](#)
- [SMS pesan teks MFA](#)
- [Token perangkat lunak TOTP MFA](#)

### Prasyarat

Sebelum Anda mengatur MFA, pertimbangkan hal berikut:

- Saat Anda mengaktifkan MFA di kumpulan pengguna dan memilih pesan teks SMS sebagai faktor kedua, Anda dapat mengirim pesan SMS ke atribut nomor telepon yang belum Anda verifikasi di Amazon Cognito. Setelah pengguna Anda menyelesaikan SMS MFA, Amazon Cognito menyetel atributnya. `phone_number_verified true`
- Jika akun Anda berada di kotak pasir SMS di Wilayah AWS yang berisi sumber daya Amazon Simple Notification Service (Amazon SNS) untuk kumpulan pengguna Anda, Anda harus memverifikasi nomor telepon di Amazon SNS sebelum dapat mengirim pesan SMS. Untuk informasi selengkapnya, lihat [Pengaturan pesan SMS untuk kolam pengguna Amazon Cognito](#).



- Fitur keamanan tingkat lanjut mengharuskan Anda mengaktifkan MFA dan mengaturnya sebagai opsional di konsol kumpulan pengguna Amazon Cognito. Untuk informasi selengkapnya, lihat [Menambahkan keamanan tingkat lanjut ke kumpulan pengguna](#).

## Mengkonfigurasi otentikasi multi-faktor

Anda dapat mengonfigurasi MFA di konsol Amazon Cognito.

Untuk mengonfigurasi MFA di konsol Amazon Cognito

1. Masuk ke [konsol Amazon Cognito](#).
2. Pilih Kolam Pengguna.
3. Pilih kolam pengguna yang ada dari daftar, atau [buat kolam pengguna](#).
4. Pilih tab Pengalaman masuk. Temukan otentikasi Multi-faktor dan pilih Edit
5. Pilih metode penegakan MFA yang ingin Anda gunakan dengan kumpulan pengguna Anda.

## Edit multi-factor authentication (MFA) [Info](#)

Amazon Cognito provides your app users with additional authentication factors using SMS messages and time-based one-time passwords (TOTP).

### Multi-factor authentication

Configure secure access to your app by enforcing multi-factor authentication (MFA) during the user sign-in process. MFA settings are applied to all app clients.

#### MFA enforcement [Info](#)

Require MFA -

**Recommended**

Users must provide an additional authentication factor when signing in.

Optional MFA

Users can sign in with a single authentication factor, and can choose to add additional authentication factors.

No MFA

Users can only sign in with a single authentication factor. This is the least secure option.

#### MFA methods [Info](#)

Choose the MFA methods that are allowed in your user pool. TOTP-based MFA offers a higher level of security. Recipient message and data rates apply.

Authenticator apps

Users can authenticate with a TOTP from an authenticator app such as Authy or Google Authenticator.

SMS message

Users can authenticate with a code sent by SMS message to a verified phone number. SMS messages are charged separately by Amazon SNS. [Learn more about pricing](#) [🔗](#) This option must be selected because SMS is configured.

Cancel

Save changes

- a. Membutuhkan MFA. Semua pengguna di kumpulan pengguna Anda harus masuk dengan kode SMS tambahan atau faktor kata sandi satu kali berbasis waktu (TOTP).
  - b. MFA opsional - Anda dapat memberi pengguna Anda opsi untuk mendaftarkan faktor masuk tambahan tetapi masih mengizinkan pengguna yang belum mengonfigurasi MFA untuk masuk. Jika Anda menggunakan otentikasi adaptif, pilih opsi ini. Untuk informasi selengkapnya tentang otentikasi adaptif, lihat [Menambahkan keamanan tingkat lanjut ke kumpulan pengguna](#)
  - c. Tidak ada MFA. Pengguna Anda tidak dapat mendaftarkan faktor masuk tambahan.
6. Pilih metode MFA yang Anda dukung di aplikasi Anda. Anda dapat mengatur pesan SMS atau aplikasi Authenticator yang menghasilkan total sebagai faktor kedua. Kami menyarankan Anda menerapkan MFA berbasis TOTP sehingga pemulihan akun dapat menggunakan pesan SMS.
  7. Jika Anda menggunakan pesan teks SMS sebagai faktor kedua dan Anda belum mengonfigurasi peran IAM untuk digunakan dengan Amazon Simple Notification Service (Amazon SNS) untuk

pesan SMS, buat satu di konsol. Di tab Pesan untuk kumpulan pengguna Anda, cari SMS dan pilih Edit. Anda juga dapat menggunakan peran yang ada yang memungkinkan Amazon Cognito mengirim pesan SMS ke pengguna untuk Anda. Untuk informasi lebih lanjut, lihat [Peran IAM](#).

8. Pilih Simpan perubahan.

## SMS pesan teks MFA

Saat pengguna masuk dengan MFA diaktifkan, mereka terlebih dahulu memasukkan dan mengirimkan nama pengguna dan kata sandi mereka. Aplikasi klien menerima respons `getMFA` yang menunjukkan di mana kode otorisasi dikirim. Aplikasi klien harus menunjukkan kepada pengguna tempat mencari kode (seperti ke nomor telepon mana kode itu dikirim). Selanjutnya, ia menyediakan formulir untuk memasukkan kode. Akhirnya, aplikasi klien mengirimkan kode untuk menyelesaikan proses masuk. Tujuannya tertutup, yang menyembunyikan semua kecuali empat digit terakhir dari nomor telepon. Jika aplikasi menggunakan UI yang dihosting Amazon Cognito, aplikasi akan menampilkan halaman bagi pengguna untuk memasukkan kode MFA.

Kode otorisasi pesan teks SMS berlaku untuk durasi sesi alur Autentikasi yang Anda tetapkan untuk klien aplikasi Anda.

Setel durasi sesi alur autentikasi di konsol Amazon Cognito di tab Integrasi aplikasi, saat Anda mengubah klien aplikasi di bawah Klien dan analitik aplikasi. Anda juga dapat mengatur durasi sesi alur autentikasi dalam permintaan `CreateUserPoolClient` atau `UpdateUserPoolClient` API. Untuk informasi selengkapnya, lihat [Alur otentikasi kumpulan pengguna](#).

Jika pengguna tidak lagi memiliki akses ke perangkat mereka di mana kode MFA pesan teks SMS dikirim, mereka harus meminta bantuan dari kantor layanan pelanggan Anda. Administrator dengan Akun AWS izin yang diperlukan dapat mengubah nomor telepon pengguna, tetapi hanya melalui AWS CLI atau API.

Ketika pengguna berhasil melewati aliran MFA pesan teks SMS, nomor telepon mereka juga ditandai sebagai terverifikasi.

### Note

SMS untuk MFA dibebankan secara terpisah. (Tidak ada biaya untuk mengirim kode verifikasi ke alamat email.) Untuk informasi selengkapnya tentang harga Amazon SNS, lihat [Harga SMS Di Seluruh Dunia](#). Untuk daftar negara saat ini yang menyediakan olahpesan SMS, lihat [Wilayah dan Negara yang Didukung](#).

### Important

Untuk memastikan bahwa pesan SMS dikirim untuk memverifikasi nomor telepon dan untuk MFA pesan teks SMS, Anda harus meminta peningkatan batas pembelanjaan dari Amazon SNS.

Amazon Cognito menggunakan Amazon SNS untuk mengirim pesan SMS ke pengguna. Jumlah pesan SMS yang dikirimkan Amazon SNS tunduk pada batasan pembelanjaan. Batas pengeluaran dapat ditentukan untuk AWS akun dan untuk pesan individual, dan batasnya hanya berlaku untuk biaya pengiriman pesan SMS.

Batas pembelanjaan default per akun (jika tidak ditentukan) adalah 1,00 USD per bulan.

Jika Anda ingin menaikkan batas, kirimkan [kasus Peningkatan Batas SNS](#) di AWS Support Pusat. Untuk Nilai batas baru, masukkan batas pengeluaran bulanan yang Anda inginkan. Di bidang Deskripsi Kasus Penggunaan, jelaskan bahwa Anda meminta peningkatan batas pengeluaran bulanan SMS.

Untuk menambahkan MFA ke kolam pengguna Anda, lihat [Menambahkan MFA ke kumpulan pengguna](#). Untuk informasi selengkapnya tentang pesan SMS dengan Amazon SNS di kumpulan pengguna, lihat. [Pengaturan pesan SMS untuk kolam pengguna Amazon Cognito](#)

## Token perangkat lunak TOTP MFA

Saat Anda mengatur MFA token perangkat lunak TOTP di kumpulan pengguna Anda, pengguna Anda masuk dengan nama pengguna dan kata sandi, lalu menggunakan TOTP untuk menyelesaikan otentikasi. Setelah pengguna Anda menetapkan dan memverifikasi nama pengguna dan kata sandi, mereka dapat mengaktifkan token perangkat lunak TOTP untuk MFA. Jika aplikasi Anda menggunakan UI yang dihosting Amazon Cognito untuk login pengguna, pengguna Anda akan mengirimkan nama pengguna dan kata sandi mereka, lalu mengirimkan kata sandi TOTP pada halaman login tambahan.

Anda dapat mengaktifkan TOTP MFA untuk kumpulan pengguna di konsol Amazon Cognito, atau Anda dapat menggunakan operasi API Amazon Cognito. Di tingkat kumpulan pengguna, Anda dapat menelepon [SetUserPoolMfaConfig](#) untuk mengkonfigurasi MFA dan mengaktifkan TOTP MFA.

### Note

Jika Anda belum mengaktifkan MFA token perangkat lunak TOTP untuk kumpulan pengguna, Amazon Cognito tidak dapat menggunakan token untuk mengaitkan atau memverifikasi

pengguna. Dalam hal ini, pengguna menerima `SoftwareTokenMFANotFoundException` pengecualian dengan deskripsi `Software Token MFA has not been enabled by the userPool`. Jika Anda menonaktifkan token perangkat lunak MFA untuk kumpulan pengguna nanti, pengguna yang sebelumnya mengaitkan dan memverifikasi token TOTP dapat terus menggunakannya untuk MFA.

Mengonfigurasi TOTP untuk pengguna Anda adalah proses multi-langkah di mana pengguna Anda menerima kode rahasia yang divalidasi dengan memasukkan kata sandi satu kali. Selanjutnya, Anda dapat mengaktifkan TOTP MFA untuk pengguna Anda atau mengatur TOTP sebagai metode MFA pilihan untuk pengguna Anda.

Saat mengonfigurasi kumpulan pengguna agar mewajibkan TOTP MFA dan pengguna mendaftar ke aplikasi Anda di UI yang dihosting, Amazon Cognito akan mengotomatiskan proses pengguna. Amazon Cognito meminta pengguna Anda untuk memilih metode MFA, menampilkan kode QR untuk menyiapkan aplikasi autentikator mereka, dan memverifikasi pendaftaran MFA mereka. Di kumpulan pengguna tempat Anda mengizinkan pengguna untuk memilih antara SMS dan TOTP MFA, Amazon Cognito juga memberi pengguna Anda pilihan metode. Untuk informasi selengkapnya tentang pengalaman pendaftaran UI yang dihosting, lihat [Cara mendaftar akun baru di UI yang dihosting Amazon Cognito](#).

#### Important

Bila Anda memiliki ACL AWS WAF web yang terkait dengan kumpulan pengguna, dan aturan di ACL web Anda menyajikan CAPTCHA, ini dapat menyebabkan kesalahan yang tidak dapat dipulihkan dalam pendaftaran TOTP UI yang dihosting. Untuk membuat aturan yang memiliki tindakan CAPTCHA dan tidak memengaruhi TOTP UI yang dihosting, lihat [Mengonfigurasi ACL AWS WAF web Anda untuk MFA TOTP UI yang dihosting](#) Untuk informasi selengkapnya tentang ACL AWS WAF web dan Amazon Cognito, lihat [Mengaitkan ACL AWS WAF web dengan kumpulan pengguna](#)

[Untuk mengimplementasikan TOTP MFA di UI khusus tempat Anda menggunakan Amazon Cognito API, lihat. Mengonfigurasi MFA untuk pengguna di API kumpulan pengguna Amazon Cognito](#)

Untuk menambahkan MFA ke kolam pengguna Anda, lihat [Menambahkan MFA ke kumpulan pengguna](#).

Pertimbangan dan batasan TOTP MFA

1. Amazon Cognito mendukung token perangkat lunak MFA melalui aplikasi autentikator yang menghasilkan kode TOTP. Amazon Cognito tidak mendukung MFA berbasis perangkat keras.
2. Ketika kumpulan pengguna Anda memerlukan TOTP untuk pengguna yang belum mengonfigurasinya, pengguna Anda menerima token akses satu kali yang dapat digunakan aplikasi Anda untuk mengaktifkan TOTP MFA bagi pengguna. Upaya masuk berikutnya gagal hingga pengguna Anda mendaftarkan faktor masuk TOTP tambahan.
  - Pengguna yang mendaftar di kumpulan pengguna Anda dengan operasi `SignUp` API atau melalui UI yang dihosting akan menerima token satu kali saat pengguna menyelesaikan pendaftaran.
  - Setelah Anda membuat pengguna, dan pengguna menetapkan kata sandi awal mereka, Amazon Cognito mengeluarkan token satu kali dari UI yang dihosting ke pengguna. Jika Anda menetapkan kata sandi permanen untuk pengguna, Amazon Cognito mengeluarkan token satu kali saat pengguna pertama kali masuk.
  - Amazon Cognito tidak mengeluarkan token satu kali ke pengguna yang dibuat administrator yang masuk dengan operasi atau API. [InitiateAuthAdminInitiateAuth](#) Setelah pengguna Anda berhasil dalam tantangan untuk mengatur kata sandi awal mereka, atau jika Anda menetapkan kata sandi permanen untuk pengguna, Amazon Cognito segera menantang pengguna untuk mengatur MFA.
3. Jika pengguna di kumpulan pengguna yang memerlukan MFA telah menerima token akses satu kali tetapi belum menyiapkan TOTP MFA, pengguna tidak dapat masuk dengan UI yang dihosting sampai mereka menyiapkan MFA. Alih-alih token akses, Anda dapat menggunakan nilai `session respons` dari `MFA_SETUP` tantangan ke [InitiateAuth](#) atau [AdminInitiateAuth](#) dalam [AssociateSoftwareToken](#) permintaan.
4. Jika pengguna Anda telah menyiapkan TOTP, mereka dapat menggunakannya untuk MFA, bahkan jika Anda menonaktifkan TOTP untuk kumpulan pengguna nanti.
5. Amazon Cognito hanya menerima TOTP dari aplikasi autentikator yang menghasilkan kode dengan fungsi hash SHA-1. Kode yang dihasilkan dengan hashing SHA-256 mengembalikan kesalahan. `Code mismatch`

## Mengonfigurasi MFA untuk pengguna di API kumpulan pengguna Amazon Cognito

Saat pengguna pertama kali masuk, aplikasi Anda menggunakan token akses satu kali untuk membuat kunci pribadi TOTP dan menyajikannya kepada pengguna Anda dalam format teks atau kode QR. Pengguna Anda mengonfigurasi aplikasi autentikatornya dan menyediakan TOTP untuk

upaya masuk berikutnya. Aplikasi Anda atau UI yang dihosting menyajikan TOTP ke Amazon Cognito dalam respons tantangan MFA.

## Topik

- [Kaitkan token perangkat lunak TOTP](#)
- [Verifikasi token TOTP](#)
- [Masuk dengan TOTP MFA](#)
- [Hapus token TOTP](#)

## Kaitkan token perangkat lunak TOTP

Untuk mengaitkan token TOTP, kirimkan kode rahasia kepada pengguna Anda yang harus mereka validasi dengan kata sandi satu kali. Mengaitkan token membutuhkan tiga langkah.

1. Saat pengguna Anda memilih token perangkat lunak TOTP MFA, panggil [AssociateSoftwareToken](#) untuk mengembalikan kode kunci rahasia bersama unik yang dihasilkan untuk akun pengguna. Anda dapat mengotorisasi AssociateSoftwareToken dengan token akses atau string sesi.
2. Aplikasi Anda menyajikan kunci pribadi kepada pengguna, atau kode QR yang Anda hasilkan dari kunci pribadi. Pengguna Anda harus memasukkan kunci ke dalam aplikasi penghasil total seperti Google Authenticator. Anda dapat menggunakan [libqrencode untuk menghasilkan kode QR](#).
3. Pengguna Anda memasukkan kunci, atau memindai kode QR ke aplikasi autentikator seperti Google Authenticator, dan aplikasi mulai menghasilkan kode.

## Verifikasi token TOTP

Selanjutnya, verifikasi token TOTP. Minta kode sampel dari pengguna Anda dan berikan ke layanan Amazon Cognito untuk mengonfirmasi bahwa pengguna berhasil membuat kode TOTP, sebagai berikut.

1. Aplikasi Anda meminta pengguna Anda untuk kode untuk menunjukkan bahwa mereka telah menyiapkan aplikasi autentikator mereka dengan benar.
2. Aplikasi autentikator pengguna menampilkan kata sandi sementara. Aplikasi autentikator mendasarkan kata sandi pada kunci rahasia yang Anda berikan kepada pengguna.
3. Pengguna Anda memasukkan kata sandi sementara mereka. Aplikasi Anda meneruskan kata sandi sementara ke Amazon Cognito dalam permintaan [VerifySoftwareToken](#) API.

4. Amazon Cognito telah mempertahankan kunci rahasia yang terkait dengan pengguna, dan menghasilkan TOTP dan membandingkannya dengan yang disediakan pengguna Anda. Jika cocok, `VerifySoftwareToken` mengembalikan SUCCESS respons.
5. Amazon Cognito mengaitkan faktor TOTP dengan pengguna.
6. Jika `VerifySoftwareToken` operasi mengembalikan ERROR respons, pastikan jam pengguna sudah benar dan belum melebihi jumlah percobaan ulang maksimum. Amazon Cognito menerima token TOTP yang berada dalam waktu 30 detik sebelum atau sesudah upaya, untuk memperhitungkan kemiringan jam kecil. Setelah Anda menyelesaikan masalah, coba `VerifySoftwareToken` operasi lagi.

## Masuk dengan TOTP MFA

Pada titik ini, pengguna Anda masuk dengan kata sandi satu kali berbasis waktu. Prosesnya adalah sebagai berikut.

1. Pengguna Anda memasukkan nama pengguna dan kata sandi mereka untuk masuk ke aplikasi klien Anda.
2. Tantangan MFA TOTP dipanggil, dan pengguna Anda diminta oleh aplikasi Anda untuk memasukkan kata sandi sementara.
3. Pengguna Anda mendapatkan kata sandi sementara dari aplikasi penghasil TOTP terkait.
4. Pengguna Anda memasukkan kode TOTP ke aplikasi klien Anda. Aplikasi Anda akan memberitahukan layanan Amazon Cognito untuk memverifikasinya. Untuk setiap login, [RespondToAuthChallenge](#) harus dipanggil untuk mendapatkan respons terhadap tantangan otentikasi TOTP yang baru.
5. Jika token diverifikasi oleh Amazon Cognito, masuk berhasil dan pengguna Anda melanjutkan dengan alur autentikasi.

## Hapus token TOTP

Terakhir, aplikasi Anda harus mengizinkan pengguna untuk menonaktifkan konfigurasi TOTP mereka. Saat ini, Anda tidak dapat menghapus token perangkat lunak TOTP pengguna. Untuk mengganti token perangkat lunak pengguna Anda, kaitkan dan verifikasi token perangkat lunak baru. Untuk menonaktifkan TOTP MFA bagi pengguna, [SetUserhubungi](#) MFAResponse untuk memodifikasi pengguna Anda agar tidak menggunakan MFA, atau hanya SMS MFA.



1. Buat antarmuka di aplikasi Anda untuk pengguna yang ingin mengatur ulang MFA. Minta pengguna di antarmuka ini untuk memasukkan kata sandi mereka.
2. [Jika Amazon Cognito mengembalikan tantangan MFA TOTP, perbarui preferensi MFA pengguna Anda dengan MFAResponse.SetUser](#)
3. Di aplikasi Anda, komunikasikan kepada pengguna bahwa mereka telah menonaktifkan MFA dan minta mereka untuk masuk lagi.

Mengonfigurasi ACL AWS WAF web Anda untuk MFA TOTP UI yang dihosting

Bila Anda memiliki ACL AWS WAF web yang terkait dengan kumpulan pengguna, dan aturan di ACL web Anda menyajikan CAPTCHA, ini dapat menyebabkan kesalahan yang tidak dapat dipulihkan dalam pendaftaran TOTP UI yang dihosting. AWS WAF Aturan CAPTCHA hanya memengaruhi TOTP MFA di UI yang dihosting dengan cara ini. SMS MFA tidak terpengaruh.

Amazon Cognito menampilkan kesalahan berikut ketika aturan CAPTCHA Anda tidak mengizinkan pengguna menyelesaikan penyiapan TOTP MFA.

Permintaan tidak diizinkan karena captcha WAF.

Kesalahan ini terjadi saat AWS WAF meminta CAPTCHA sebagai respons

[AssociateSoftwareToken](#) dan permintaan [VerifySoftwareToken](#) API yang dibuat oleh kumpulan pengguna Anda di latar belakang. Untuk membuat aturan yang memiliki tindakan CAPTCHA dan tidak memengaruhi TOTP UI yang dihosting, kecualikan nilai `x-amzn-cognito-operation-name` header `AssociateSoftwareToken` dan `VerifySoftwareToken` dari tindakan CAPTCHA dalam aturan Anda.

Screenshot berikut menunjukkan contoh AWS WAF aturan yang menerapkan tindakan CAPTCHA untuk semua permintaan yang tidak memiliki nilai `x-amzn-cognito-operation-name` header atau `AssociateSoftwareToken` `VerifySoftwareToken`

## If a request matches all the statements (AND)

### NOT Statement 1

Field to match

Single header (x-amzn-cognito-operation-name)

Positional constraint

Exactly matches string

Search string

AssociateSoftwareToken

Text transformations

- None (Priority 0)

AND

### NOT Statement 2

Field to match

Single header (x-amzn-cognito-operation-name)

Positional constraint

Exactly matches string

Search string

VerifySoftwareToken

Text transformations

- None (Priority 0)

## Then

### Action

The action to take when a web request matches the rule statement.

Untuk informasi selengkapnya tentang ACL AWS WAF web dan Amazon Cognito, lihat [Mengaitkan ACL AWS WAF web dengan kumpulan pengguna](#)

## Menambahkan keamanan tingkat lanjut ke kumpulan pengguna

Setelah membuat kolam pengguna, Anda memiliki akses ke Keamanan lanjutan pada bar navigasi di konsol Amazon Cognito. Anda dapat mengaktifkan fitur keamanan lanjutan kolam pengguna, dan menyesuaikan tindakan yang diambil sebagai respons terhadap risiko yang berbeda. Atau Anda dapat menggunakan mode audit untuk mengumpulkan metrik risiko yang terdeteksi tanpa menerapkan mitigasi keamanan apa pun. Dalam mode audit, fitur keamanan lanjutan mempublikasikan metrik ke Amazon CloudWatch. Anda dapat melihat metrik keamanan lanjutan setelah Amazon Cognito menghasilkan peristiwa keamanan lanjutan pertamanya. Lihat [Melihat metrik keamanan tingkat lanjut](#).

Fitur keamanan lanjutan termasuk deteksi kredensial yang dikompromikan dan otentikasi adaptif.

### Kredensial yang dikompromikan

Pengguna menggunakan kembali kata sandi untuk beberapa akun pengguna. Fitur kredensial yang dikompromikan dari Amazon Cognito mengkompilasi data dari kebocoran publik nama pengguna dan kata sandi, dan membandingkan kredensial pengguna Anda dengan daftar kredensial yang bocor. Deteksi kredensial yang dikompromikan juga memeriksa kata sandi yang umum ditebak.

Anda dapat memilih tindakan pengguna yang meminta pemeriksaan kredensial yang dikompromikan, dan tindakan yang Anda ingin dilakukan Amazon Cognito sebagai tanggapan. Untuk peristiwa login, pendaftaran, dan perubahan kata sandi, Amazon Cognito dapat Memblokir proses masuk, atau Izinkan masuk. Dalam kedua kasus tersebut, Amazon Cognito menghasilkan log aktivitas pengguna, tempat Anda dapat menemukan informasi lebih lanjut tentang acara tersebut.

### Otentikasi adaptif

Amazon Cognito dapat meninjau informasi lokasi dan perangkat dari permintaan masuk pengguna Anda dan menerapkan respons otomatis untuk mengamankan akun pengguna di kumpulan pengguna Anda dari aktivitas mencurigakan.

Saat Anda mengaktifkan keamanan tingkat lanjut, Amazon Cognito memberikan skor risiko untuk aktivitas pengguna. Anda dapat menetapkan respons otomatis terhadap aktivitas mencurigakan: Anda dapat Memerlukan MFA, Memblokir proses masuk, atau hanya mencatat detail aktivitas dan skor risiko. Anda juga dapat secara otomatis mengirim pesan email yang memberi tahu pengguna

Anda tentang aktivitas mencurigakan sehingga mereka dapat mengatur ulang kata sandi mereka atau mengambil tindakan mandiri lainnya.

## Akses kustomisasi token

Saat mengaktifkan fitur keamanan lanjutan, Anda dapat mengonfigurasi kumpulan pengguna untuk menerima respons terhadap peristiwa pemicu Lambda versi 2. Dengan versi 2, Anda dapat menyesuaikan cakupan dan klaim lainnya dalam token akses. Ini meningkatkan kemampuan Anda untuk membuat hasil otorisasi yang fleksibel saat pengguna Anda mengautentikasi. Untuk informasi selengkapnya, lihat [Menyesuaikan token akses](#).

## Topik

- [Pertimbangan dan batasan](#)
- [Prasyarat](#)
- [Mengkonfigurasi fitur keamanan tingkat lanjut](#)
- [Memeriksa kredensial yang dikompromikan](#)
- [Menggunakan otentikasi adaptif](#)
- [Melihat metrik keamanan tingkat lanjut](#)
- [Mengaktifkan keamanan lanjutan kumpulan pengguna dari aplikasi Anda](#)

## Pertimbangan dan batasan

- Harga tambahan berlaku untuk fitur keamanan lanjutan Amazon Cognito. Lihat [Halaman harga Amazon Cognito](#).
- Amazon Cognito mendukung autentikasi adaptif dan deteksi kredensial-kompromi dengan alur otentikasi standar berikut: `USER_PASSWORD_AUTH`, `ADMIN_USER_PASSWORD_AUTH`, `USER_SRP_AUTH`. Anda tidak dapat menggunakan keamanan tingkat lanjut dengan `CUSTOM_AUTH` alur dan [Tantangan otentikasi kustom pemicu Lambda](#), atau dengan login gabungan.
- Dengan fitur keamanan canggih Amazon Cognito dalam mode fungsi penuh, Anda dapat membuat alamat IP Selalu blokir dan Selalu izinkan pengecualian. Sesi dari alamat IP pada daftar pengecualian Selalu blokir tidak ditetapkan tingkat risiko dengan autentikasi adaptif, dan tidak dapat masuk ke kumpulan pengguna Anda.
- Permintaan yang diblokir dari alamat IP pada daftar pengecualian Selalu blokir di kumpulan pengguna Anda berkontribusi pada [kuota tingkat permintaan](#) untuk kumpulan pengguna Anda. Fitur keamanan canggih Amazon Cognito tidak mencegah serangan penolakan layanan

terdistribusi (DDoS). Untuk menerapkan pertahanan terhadap serangan volumetrik di kumpulan pengguna Anda, tambahkan AWS WAF ACL web. Untuk informasi selengkapnya, lihat [Mengaitkan ACL AWS WAF web dengan kumpulan pengguna](#).

- Hibah kredensi klien dimaksudkan untuk otorisasi machine-to-machine (M2M) tanpa koneksi ke akun pengguna. Fitur keamanan tingkat lanjut hanya memantau akun pengguna dan kata sandi di kumpulan pengguna Anda. Untuk menerapkan fitur keamanan dengan aktivitas M2M Anda, pertimbangkan kemampuan AWS WAF untuk memantau tingkat permintaan dan konten. Untuk informasi selengkapnya, lihat [Mengaitkan ACL AWS WAF web dengan kumpulan pengguna](#).

## Prasyarat

Sebelum memulai, Anda perlu melakukan hal berikut:

- Kolam pengguna dengan klien aplikasi. Untuk informasi selengkapnya, lihat [Memulai dengan kumpulan pengguna](#).
- Atur Autentikasi Multi-Faktor (MFA) ke Opsional di konsol Amazon Cognito untuk menggunakan fitur autentikasi adaptif berbasis risiko. Untuk informasi selengkapnya, lihat [Menambahkan MFA ke kumpulan pengguna](#).
- Jika Anda menggunakan notifikasi email, buka [konsol Amazon SES](#) untuk mengonfigurasi dan memverifikasi alamat email atau domain yang akan digunakan dengan notifikasi email Anda. Untuk informasi lebih lanjut tentang Amazon SES, lihat [Memverifikasi Identitas di Amazon SES](#).

## Mengkonfigurasi fitur keamanan tingkat lanjut

Anda dapat mengonfigurasi fitur keamanan lanjutan Amazon Cognito di AWS Management Console

Untuk mengonfigurasi keamanan lanjutan untuk kolam pengguna

1. Masuk ke [Konsol Amazon Cognito](#). Jika diminta, masukkan AWS kredensial Anda.
2. Pilih Kolam Pengguna.
3. Pilih kolam pengguna yang ada dari daftar, atau [buat kolam pengguna](#).
4. Pilih tab Integrasi aplikasi. Temukan Keamanan tingkat lanjut dan pilih Aktifkan. Jika Anda mengaktifkan keamanan lanjutan sebelumnya, pilih Edit.
5. Pilih Fungsi penuh untuk mengonfigurasi respons keamanan lanjutan ke kredensial yang dikompromikan dan otentikasi adaptif. Pilih Audit hanya untuk mengumpulkan informasi dan mengirim data kumpulan pengguna ke CloudWatch. Harga keamanan tingkat lanjut berlaku

dalam mode Audit saja dan fungsi penuh. Untuk informasi selengkapnya, lihat [Harga Amazon Cognito](#).

Sebaiknya pertahankan fitur keamanan lanjutan dalam mode audit selama dua minggu sebelum mengaktifkan tindakan. Selama waktu ini, Amazon Cognito dapat mempelajari pola penggunaan pengguna aplikasi Anda.

6. Jika Anda memilih Audit saja, pilih Simpan perubahan. Jika Anda memilih Fungsi penuh:
  - a. Pilih apakah Anda akan mengambil tindakan Kustom atau menggunakan atau Cognito default untuk menanggapi dugaan kredensial Terkompromi. Cognito default adalah:
    - i. Mendeteksi kredensial yang dikompromikan saat Masuk, Mendaftar, dan Perubahan Kata Sandi.
    - ii. Menanggapi kredensial yang dikompromikan dengan tindakan Blokir login.
  - b. Jika Anda memilih Tindakan kustom untuk kredensial yang dikompromikan, pilih tindakan kumpulan pengguna yang akan digunakan Amazon Cognito untuk deteksi Peristiwa dan tanggapan kredensial yang dikompromikan yang Anda ingin dilakukan Amazon Cognito. Anda dapat Memblokir proses masuk atau Izinkan masuk dengan kredensial yang dicurigai disusupi.
  - c. Pilih cara menanggapi upaya masuk berbahaya di bawah Autentikasi adaptif. Pilih apakah Anda akan mengambil tindakan Kustom atau menggunakan atau Cognito default untuk menanggapi aktivitas berbahaya yang dicurigai. Saat Anda memilih Cognito default, Amazon Cognito memblokir proses masuk di semua tingkat risiko dan tidak memberi tahu pengguna.
  - d. Jika Anda memilih Tindakan kustom untuk otentikasi Adaptif, pilih Tindakan respons risiko otomatis yang akan diambil Amazon Cognito sebagai respons terhadap risiko yang terdeteksi berdasarkan tingkat keparahan. Ketika Anda menetapkan respons terhadap tingkat risiko, Anda tidak dapat menetapkan respons yang kurang membatasi ke tingkat risiko yang lebih tinggi. Anda dapat menetapkan tanggapan berikut untuk tingkat risiko:
    - i. Izinkan masuk - Jangan mengambil tindakan pencegahan.
    - ii. MFA opsional - Jika pengguna memiliki MFA yang dikonfigurasi, Amazon Cognito akan selalu meminta pengguna untuk memberikan SMS tambahan atau faktor kata sandi satu kali berbasis waktu (TOTP) saat mereka masuk. Jika pengguna tidak memiliki MFA yang dikonfigurasi, mereka dapat terus masuk secara normal.
    - iii. Memerlukan MFA - Jika pengguna memiliki MFA yang dikonfigurasi, Amazon Cognito akan selalu meminta pengguna untuk memberikan SMS atau faktor TOTP tambahan

saat mereka masuk. Jika pengguna tidak memiliki MFA yang dikonfigurasi, Amazon Cognito akan meminta mereka untuk mengatur MFA. Sebelum Anda secara otomatis memerlukan MFA untuk pengguna Anda, konfigurasi mekanisme di aplikasi Anda untuk menangkap nomor telepon untuk SMS MFA, atau untuk mendaftarkan aplikasi autentikator untuk TOTP MFA.

- iv. Blokir masuk - Cegah pengguna masuk.
  - v. Beri tahu pengguna - Kirim pesan email ke pengguna dengan informasi tentang risiko yang terdeteksi Amazon Cognito dan respons yang telah Anda ambil. Anda dapat menyesuaikan template pesan email untuk pesan yang Anda kirim.
7. Jika Anda memilih Beri tahu pengguna pada langkah sebelumnya, Anda dapat menyesuaikan pengaturan pengiriman email dan templat pesan email untuk autentikasi adaptif.
- a. Di bawah Konfigurasi email, pilih Wilayah SES, DARI alamat email, DARI nama pengirim, dan alamat email BALAS-KE yang ingin Anda gunakan dengan autentikasi adaptif. Untuk informasi selengkapnya tentang mengintegrasikan pesan email kumpulan pengguna Anda dengan Amazon Simple Email Service, lihat [Pengaturan email untuk kumpulan pengguna Amazon Cognito](#).

### Adaptive authentication messages

Customize the messages sent to users when adaptive authentication triggers a notification. Adaptive authentication messages use [Amazon SES](#).

#### Email configuration

Configure the [Amazon SES](#) verified identity used to send adaptive authentication messages. [Learn more](#)

**SES Region** [Info](#)  
Choose an AWS Region to use with SES in this user pool. For best performance, you should configure SES and your user pool in the same Region.

US East (N. Virginia) ▼

**FROM email address** [Info](#)  
Choose an email address that you have verified with Amazon SES.

▼

**FROM sender name - optional** [Info](#)  
Enter a friendly name for the email sender in the format "John Stiles <johnstiles@example.com>."

**REPLY-TO email address - optional** [Info](#)  
If you set an invalid reply-to address, sending restrictions may be imposed on your account.

▼ **Email templates**

#### Risk detected, sign-in allowed

**Email subject** [Reset to default](#)

New sign-in attempt

**Email message - Text** [Reset to default](#)    **Email message - HTML** [Reset to default](#)

We observed an unrecognized sign-in to your     <!DOCTYPE html>

- b. Perluas template Email untuk menyesuaikan notifikasi autentikasi adaptif dengan pesan email versi HTML dan teks biasa. Untuk mempelajari lebih lanjut tentang templat pesan email, lihat [Template pesan](#).
8. Perluas pengecualian alamat IP untuk membuat daftar Always-allow atau Always-block dari rentang alamat IPv4 atau IPv6 yang akan selalu diizinkan atau diblokir, terlepas dari penilaian risiko keamanan tingkat lanjut. Tentukan rentang alamat IP di [Notasi CIDR](#) (seperti 192.168.100.0/24).
9. Pilih Simpan perubahan.



## Memeriksa kredensial yang dikompromikan

Amazon Cognito dapat mendeteksi apakah nama pengguna dan kata sandi pengguna telah disusupi di tempat lain. Hal ini dapat terjadi ketika pengguna menggunakan kembali kredensi di lebih dari satu situs, atau ketika mereka menggunakan kata sandi yang tidak aman. Amazon Cognito memeriksa pengguna lokal yang masuk dengan nama pengguna dan kata sandi, di UI yang dihosting, dan dengan Amazon Cognito API. Pengguna lokal ada secara eksklusif di direktori kumpulan pengguna Anda tanpa federasi melalui iDP eksternal.

Dari Keamanan lanjutan di tab Integrasi aplikasi di konsol Amazon Cognito, Anda dapat mengonfigurasi kredensial yang dikompromikan. Konfigurasi deteksi Peristiwa untuk memilih peristiwa pengguna yang ingin Anda pantau untuk kredensial yang disusupi. Konfigurasi tanggapan kredensial yang dikompromikan untuk memilih apakah akan mengizinkan atau memblokir pengguna jika kredensial yang disusupi terdeteksi. Amazon Cognito dapat memeriksa kredensial yang disusupi selama proses masuk, pendaftaran, dan perubahan kata sandi.

Saat memilih Izinkan masuk, Anda dapat meninjau CloudWatch Log Amazon untuk memantau evaluasi yang dibuat Amazon Cognito pada peristiwa pengguna. Untuk informasi selengkapnya, lihat [Melihat metrik keamanan tingkat lanjut](#). Saat Anda memilih Blokir proses masuk, Amazon Cognito mencegah proses masuk oleh pengguna yang menggunakan kredensi yang disusupi. Saat Amazon Cognito memblokir proses masuk untuk pengguna, Amazon Cognito akan menyetel akses pengguna. [UserStatus](#)RESET\_REQUIRED Pengguna dengan RESET\_REQUIRED status harus mengubah kata sandi mereka sebelum mereka dapat masuk lagi.

### Note

Saat ini, Amazon Cognito tidak memeriksa kredensial yang dikompromikan untuk operasi masuk dengan alur Kata Sandi Jarak Jauh Aman (SRP). SRP mengirimkan bukti kata sandi yang di-hash saat masuk. Amazon Cognito tidak memiliki akses ke kata sandi secara internal, sehingga hanya dapat mengevaluasi kata sandi yang diteruskan klien Anda dalam teks biasa. Amazon Cognito memeriksa login yang menggunakan [AdminInitiateAuth](#) API dengan ADMIN\_USER\_PASSWORD\_AUTH flow, dan API dengan flow, untuk kredensial yang [InitiateAuth](#) USER\_PASSWORD\_AUTH disusupi.

Untuk menambahkan perlindungan kredensial yang dikompromikan ke kolam pengguna Anda, lihat [Menambahkan keamanan tingkat lanjut ke kumpulan pengguna](#).

## Menggunakan otentikasi adaptif

Dengan autentikasi adaptif, Anda dapat mengonfigurasi kumpulan pengguna untuk memblokir login yang mencurigakan atau menambahkan otentikasi faktor kedua sebagai respons terhadap peningkatan tingkat risiko. Untuk setiap upaya masuk, Amazon Cognito menghasilkan skor risiko untuk seberapa besar kemungkinan permintaan masuk berasal dari sumber yang dikompromikan. Skor risiko ini didasarkan pada faktor-faktor yang mencakup informasi perangkat dan pengguna. Autentikasi adaptif dapat mengaktifkan atau memerlukan otentikasi multi-faktor (MFA) untuk pengguna di kumpulan pengguna Anda saat Amazon Cognito mendeteksi risiko dalam sesi pengguna, dan pengguna belum memilih metode MFA. Ketika Anda mengaktifkan MFA untuk pengguna, mereka selalu menerima tantangan untuk menyediakan atau mengatur faktor kedua selama otentikasi, terlepas dari bagaimana Anda mengonfigurasi otentikasi adaptif. Dari sudut pandang pengguna, aplikasi Anda menawarkan untuk membantu mereka menyiapkan MFA, dan secara opsional Amazon Cognito mencegah mereka masuk lagi hingga mereka mengonfigurasi faktor tambahan.

Amazon Cognito menerbitkan upaya masuk, tingkat risikonya, dan tantangan yang gagal ke Amazon CloudWatch. Untuk informasi selengkapnya, lihat [Melihat metrik keamanan tingkat lanjut](#).

Untuk menambahkan autentikasi adaptif ke kolam pengguna Anda, lihat [Menambahkan keamanan tingkat lanjut ke kumpulan pengguna](#).

### Topik

- [Ikhtisar otentikasi adaptif](#)
- [Menambahkan perangkat pengguna dan data sesi ke permintaan API](#)
- [Melihat riwayat acara pengguna](#)
- [Memberikan umpan balik acara](#)
- [Mengirim pesan notifikasi](#)

### Ikhtisar otentikasi adaptif


Dari Keamanan tingkat lanjut di tab Integrasi aplikasi di konsol Amazon Cognito, Anda dapat memilih pengaturan untuk otentikasi adaptif, termasuk tindakan apa yang harus diambil pada tingkat risiko yang berbeda dan penyesuaian pesan notifikasi kepada pengguna. Anda dapat menetapkan konfigurasi keamanan lanjutan global ke semua klien aplikasi Anda, tetapi menerapkan konfigurasi tingkat klien ke klien aplikasi individual.

Autentikasi adaptif Amazon Cognito menetapkan salah satu tingkat risiko berikut untuk setiap sesi pengguna: Tinggi, Sedang, Rendah, atau Tanpa Risiko.

Pertimbangkan opsi Anda dengan cermat saat mengubah metode Penegakan dari hanya Audit ke Fungsi Penuh. Respons otomatis yang Anda terapkan pada tingkat risiko memengaruhi tingkat risiko yang diberikan Amazon Cognito ke sesi pengguna berikutnya dengan karakteristik yang sama. Misalnya, setelah Anda memilih untuk tidak mengambil tindakan, atau Izinkan, sesi pengguna yang awalnya dievaluasi Amazon Cognito berisiko tinggi, Amazon Cognito menganggap sesi serupa memiliki risiko lebih rendah.

Untuk setiap tingkat risiko, Anda dapat memilih dari opsi berikut:

Opsi	Tindakan
Izinkan	Pengguna dapat masuk tanpa faktor tambahan.
MFA Opsional	Pengguna yang memiliki faktor kedua yang dikonfigurasi harus menyelesaikan tantangan faktor kedua untuk masuk. Nomor telepon untuk SMS dan token perangkat lunak TOTP adalah faktor kedua yang tersedia. Pengguna tanpa faktor kedua yang dikonfigurasi dapat masuk hanya dengan satu set kredensial.
Memerlukan MFA	Pengguna yang memiliki faktor kedua yang dikonfigurasi harus menyelesaikan tantangan faktor kedua untuk masuk. Amazon Cognito memblokir proses masuk untuk pengguna yang tidak memiliki faktor kedua yang dikonfigurasi.
Blokir	Amazon Cognito memblokir semua upaya masuk pada tingkat risiko yang ditentukan.

 Note

Anda tidak perlu memverifikasi nomor telepon untuk menggunakannya untuk SMS sebagai faktor otentikasi kedua.

## Menambahkan perangkat pengguna dan data sesi ke permintaan API

Anda dapat mengumpulkan dan meneruskan informasi tentang sesi pengguna Anda ke keamanan lanjutan Amazon Cognito saat Anda menggunakan API untuk mendaftar, masuk, dan mengatur ulang kata sandi mereka. Informasi ini mencakup alamat IP pengguna Anda dan pengenalan perangkat unik.

Anda mungkin memiliki perangkat jaringan perantara antara pengguna dan Amazon Cognito, seperti layanan proxy atau server aplikasi. Anda dapat mengumpulkan data konteks pengguna dan meneruskannya ke Amazon Cognito sehingga autentikasi adaptif menghitung risiko Anda berdasarkan karakteristik titik akhir pengguna, bukan server atau proxy Anda. Jika aplikasi sisi klien Anda memanggil operasi Amazon Cognito API secara langsung, autentikasi adaptif secara otomatis mencatat alamat IP sumber. Namun, itu tidak merekam informasi perangkat lain seperti `user-agent` kecuali Anda juga mengumpulkan sidik jari perangkat.

Hasilkan data ini dengan pustaka pengumpulan data konteks Amazon Cognito dan kirimkan ke keamanan lanjutan Amazon Cognito dengan [ContextData](#) parameter dan [UserContextData](#) Pustaka pengumpulan data konteks disertakan dalam AWS SDK. Untuk informasi selengkapnya, silakan lihat [Mengintegrasikan Amazon Cognito dengan aplikasi web dan seluler](#). Anda dapat mengirimkan `ContextData` jika Anda telah mengaktifkan fitur keamanan lanjutan di kumpulan pengguna Anda. Untuk informasi selengkapnya, lihat [Mengonfigurasi fitur keamanan lanjutan](#).

Saat Anda memanggil operasi API yang diautentikasi Amazon Cognito berikut dari server aplikasi Anda, teruskan IP perangkat pengguna dalam parameter `ContextData`. Selain itu, berikan nama server, jalur server, dan data sidik jari perangkat yang dikodekan.

- [AdminInitiateAuth](#)
- [AdminRespondToAuthChallenge](#)

Saat memanggil operasi API Amazon Cognito yang tidak diautentikasi, Anda dapat mengirimkan ke fitur keamanan lanjutan Amazon `UserContextData` Cognito. Data ini mencakup sidik jari perangkat dalam `EncodedData` parameter. Anda juga dapat mengirimkan `IpAddress` parameter di `UserContextData` jika Anda memenuhi ketentuan berikut:

- Anda telah mengaktifkan fitur keamanan tingkat lanjut di kumpulan pengguna Anda. Untuk informasi selengkapnya, lihat [Mengonfigurasi fitur keamanan lanjutan](#).
- Klien aplikasi Anda memiliki rahasia klien. Untuk informasi selengkapnya, lihat [Mengonfigurasi klien aplikasi kumpulan pengguna](#).

- Anda telah mengaktifkan Terima data konteks pengguna tambahan di klien aplikasi Anda. Untuk informasi selengkapnya, lihat [Menerima data konteks pengguna tambahan \(\)AWS Management Console](#).

Aplikasi Anda dapat mengisi `UserContextData` parameter dengan data sidik jari perangkat yang dikodekan dan alamat IP perangkat pengguna dalam operasi API Amazon Cognito yang tidak diautentikasi berikut.

- [InitiateAuth](#)
- [RespondToAuthChallenge](#)
- [SignUp](#)
- [ConfirmSignUp](#)
- [ForgotPassword](#)
- [ConfirmForgotPassword](#)
- [ResendConfirmationCode](#)

### Menerima data konteks pengguna tambahan ()AWS Management Console

Kumpulan pengguna Anda menerima alamat IP dalam `UserContextData` parameter setelah Anda mengaktifkan fitur Terima data konteks pengguna tambahan. Anda tidak perlu mengaktifkan fitur ini jika:

- Pengguna Anda hanya masuk dengan operasi API yang diautentikasi seperti [AdminInitiateAuth](#), dan Anda menggunakan `ContextData` parameternya.
- Anda hanya ingin operasi API yang tidak diautentikasi mengirim sidik jari perangkat, tetapi bukan alamat IP, ke fitur keamanan lanjutan Amazon Cognito.

Perbarui klien aplikasi Anda sebagai berikut di konsol Amazon Cognito untuk menambahkan dukungan untuk data konteks pengguna tambahan.

1. Masuk ke [konsol Amazon Cognito](#).
2. Di panel navigasi, pilih Kelola Kolam Pengguna Anda, lalu pilih kolam pengguna yang ingin Anda edit.
3. Pilih tab Integrasi aplikasi.

4. Di bawah Klien dan analitik aplikasi, pilih atau buat klien aplikasi. Untuk informasi selengkapnya, lihat [Mengonfigurasi klien aplikasi kumpulan pengguna](#).
5. Pilih Edit dari wadah informasi klien App.
6. Di setelan autentikasi lanjutan untuk klien aplikasi Anda, pilih Terima data konteks pengguna tambahan.
7. Pilih Simpan perubahan.

Untuk mengonfigurasi klien aplikasi agar menerima data konteks pengguna di Amazon Cognito API, setel `EnablePropagateAdditionalUserContextData` ke `true` dalam permintaan [CreateUserPoolClient](#) atau [UpdateUserPoolClient](#) permintaan. Untuk informasi tentang cara mengaktifkan keamanan lanjutan dari web atau aplikasi seluler, lihat [Mengaktifkan keamanan lanjutan kumpulan pengguna dari aplikasi Anda](#). Saat aplikasi memanggil Amazon Cognito dari server, kumpulkan data konteks pengguna dari sisi klien. Berikut ini adalah contoh yang menggunakan metode JavaScript `getData` SDK.

```
var encodedData =  
  AmazonCognitoAdvancedSecurityData.getData(username, userPoolId, clientId);
```

Saat mendesain aplikasi untuk menggunakan autentikasi adaptif, kami sarankan Anda memasukkan SDK Amazon Cognito terbaru ke dalam aplikasi Anda.. Versi terbaru SDK mengumpulkan informasi sidik jari perangkat seperti ID perangkat, model, dan zona waktu. Untuk informasi lebih lanjut tentang Amazon Cognito SDK, lihat [Memasang SDK kolam pengguna](#). Keamanan lanjutan Amazon Cognito hanya menyimpan dan menetapkan skor risiko untuk peristiwa yang dikirimkan aplikasi Anda dalam format yang benar. Jika Amazon Cognito mengembalikan respons kesalahan, periksa apakah permintaan Anda menyertakan hash rahasia yang valid dan bahwa `IPAddress` parameternya adalah alamat IPv4 atau IPv6 yang valid.

**ContextData** dan **UserContextData** sumber daya

- AWS Amplify SDK for Android: [GetUserContextData](#)
- AWS Amplify SDK for iOS: [userContextData](#)
- JavaScript: [amazon-cognito-advanced-security-data.min.js](#)

## Melihat riwayat acara pengguna

### Note

Di konsol Amazon Cognito baru, Anda dapat melihat riwayat peristiwa pengguna di tab Pengguna.

Untuk melihat riwayat login pengguna, Anda dapat memilih pengguna dari tab Pengguna di konsol Amazon Cognito. Amazon Cognito mempertahankan riwayat peristiwa pengguna selama dua tahun.

Date (UTC)	Event	Result	Risk level	Risk decision	Challenge	IP	Device	Location	Event feedback
Jan 23, 2018 11:43:05 PM	Sign In	Pass	-	No Risk	Password:Success	52.94.36.11	Chrome, Windows 10	London	-
Jan 23, 2018 11:42:14 PM	Sign In	Pass	-	No Risk	Password:Success	52.94.36.11	Chrome, Windows 10	London	-
Jan 18, 2018 9:21:21 PM	Sign In	Fail	High	Account Takeover	Password:Success	67.132.130.174	Chrome Mobile, Android Mobile	Seattle	-
Jan 18, 2018 9:20:28 PM	Sign In	In Progress	High	Account Takeover	Password:Success	67.132.130.174	Chrome Mobile, Android Mobile	Seattle	-
Jan 18, 2018 9:18:18 PM	Sign In	Pass	-	No Risk	Password:Success	67.132.130.174	Chrome Mobile, Android Mobile	Seattle	Invalid

5 per page < 1 2 3 >

Setiap acara masuk memiliki ID acara. Acara ini juga memiliki data konteks yang sesuai, seperti lokasi, detail perangkat, dan hasil deteksi risiko. [Anda dapat melakukan kueri riwayat peristiwa pengguna dengan operasi Amazon Cognito API AdminListUserAuthEvents atau dengan AWS Command Line Interface \(AWS CLI\) dengan `admin-list-user-auth -events`.](#)

Anda juga dapat menghubungkan ID acara dengan token yang dikeluarkan Amazon Cognito pada saat merekam acara. ID dan token akses menyertakan ID peristiwa ini dalam muatannya. Amazon Cognito juga menghubungkan penggunaan token refresh dengan ID acara asli. Anda dapat melacak ID peristiwa asli kembali ke ID peristiwa login yang mengakibatkan penerbitan token Amazon Cognito. Anda dapat melacak penggunaan token dalam sistem Anda ke acara otentikasi tertentu. Untuk informasi selengkapnya, lihat [Menggunakan token dengan kolam pengguna](#).

## Memberikan umpan balik acara

Umpan balik peristiwa mempengaruhi evaluasi risiko secara real time dan meningkatkan algoritma evaluasi risiko dari waktu ke waktu. Anda dapat memberikan umpan balik tentang validitas upaya masuk melalui konsol Amazon Cognito dan operasi API.

### Note

Umpan balik acara Anda memengaruhi tingkat risiko yang diberikan Amazon Cognito ke sesi pengguna berikutnya dengan karakteristik yang sama.

Di konsol Amazon Cognito, pilih pengguna dari tab Pengguna dan pilih Berikan umpan balik peristiwa. Anda dapat meninjau detail acara dan Set sebagai valid atau Set sebagai tidak valid.

Konsol mencantumkan riwayat masuk di tab Pengguna dan grup. Jika Anda memilih entri, Anda dapat menandai acara sebagai valid atau tidak valid. Anda juga dapat memberikan umpan balik melalui operasi API kumpulan pengguna [AdminUpdateAuthEventFeedback](#), dan melalui AWS CLI perintah [admin-update-auth-event-feedback](#).

Saat Anda memilih Setel sebagai valid di konsol Amazon Cognito atau memberikan FeedbackValue nilai valid di API, Anda memberi tahu Amazon Cognito bahwa Anda mempercayai sesi pengguna di mana Amazon Cognito telah mengevaluasi beberapa tingkat risiko. Jika Anda memilih Setel sebagai tidak valid di konsol Amazon Cognito atau memberikan FeedbackValue nilai invalid di API, Anda memberi tahu Amazon Cognito bahwa Anda tidak mempercayai sesi pengguna, atau Anda tidak percaya bahwa Amazon Cognito mengevaluasi tingkat risiko yang cukup tinggi.

## Mengirim pesan notifikasi

Dengan perlindungan keamanan tingkat lanjut, Amazon Cognito dapat memberi tahu pengguna Anda tentang upaya masuk yang berisiko. Amazon Cognito juga dapat meminta pengguna untuk memilih tautan untuk menunjukkan apakah proses masuk valid atau tidak valid. Amazon Cognito menggunakan umpan balik ini untuk meningkatkan akurasi deteksi risiko untuk kumpulan pengguna Anda.

Di bagian Respons risiko otomatis pilih Beri tahu Pengguna untuk kasus berisiko rendah, sedang, atau tinggi.



Automatic risk response <a href="#">Info</a>					
Risk level	Allow sign-in	Optional MFA	Require MFA	Block sign-in	Notify user
Low risk	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="checkbox"/>
Medium risk	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="checkbox"/>
High risk	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input checked="" type="checkbox"/>

Amazon Cognito mengirimkan pemberitahuan email kepada pengguna Anda terlepas dari apakah mereka telah memverifikasi alamat email mereka.

Anda dapat menyesuaikan pesan email pemberitahuan, dan memberikan versi teks biasa dan HTML dari pesan-pesan ini. Untuk menyesuaikan notifikasi email Anda, buka template Email dari pesan autentikasi adaptif dalam konfigurasi keamanan lanjutan Anda. Untuk mempelajari selengkapnya tentang templat email, lihat [Template pesan](#).

## Melihat metrik keamanan tingkat lanjut

Amazon Cognito menerbitkan metrik untuk fitur keamanan tingkat lanjut ke akun Anda di Amazon. CloudWatch Amazon Cognito mengelompokkan metrik keamanan lanjutan berdasarkan tingkat risiko dan juga berdasarkan tingkat permintaan.

Untuk melihat metrik di konsol CloudWatch

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Pada panel navigasi, silakan pilih Metrik.
3. Pilih Amazon Cognito.
4. Pilih grup metrik gabungan, seperti Berdasarkan Klasifikasi Risiko.
5. Tab Semua metrik menampilkan semua metrik untuk pilihan tersebut. Anda dapat melakukan hal berikut:
  - Untuk menyortir tabel, gunakan judul kolomnya.
  - Untuk membuat grafik sebuah metrik, pilih kotak centang di sebelah metrik. Untuk memilih semua metrik, pilih kotak centang di baris judul tabel.
  - Untuk memfilter berdasarkan sumber daya, pilih ID sumber daya, lalu pilih Tambahkan ke pencarian.

- Untuk memfilter berdasarkan metrik, pilih nama metrik, lalu pilih Tambahkan ke pencarian.

Metrik	Deskripsi	Dimensi Metrik
CompromisedCredentialRisk	Permintaan di mana Amazon Cognito mendeteksi kredensial yang dikompromikan.	<p>Operasi: Jenis operasi. PasswordChange ,SignIn, atau SignUp satu-satunya dimensi.</p> <p>UserPoolId: Pengidentifikasi kumpulan pengguna.</p> <p>RiskLevel: tinggi (default), sedang, atau rendah.</p>
AccountTakeoverRisk	Permintaan di mana Amazon Cognito mendeteksi risiko pengambilalihan akun.	<p>Operasi: Jenis operasi. PasswordChange ,SignIn, atau SignUp satu-satunya dimensi.</p> <p>UserPoolId: Pengidentifikasi kumpulan pengguna.</p> <p>RiskLevel: tinggi, sedang, atau rendah.</p>
OverrideBlock	Meminta agar Amazon Cognito diblokir karena konfigurasi yang disediakan oleh developer.	<p>Operasi: Jenis operasi. PasswordChange ,SignIn, atau SignUp satu-satunya dimensi.</p> <p>UserPoolId: Pengidentifikasi kumpulan pengguna.</p> <p>RiskLevel: tinggi, sedang, atau rendah.</p>

Metrik	Deskripsi	Dimensi Metrik
Risiko	Permintaan bahwa Amazon Cognito ditandai sebagai berisiko.	Operasi: Jenis operasi, seperti PasswordChange, SignIn, atau SignUp.  UserPoolId: Pengidentifikasi kumpulan pengguna.
NoRisk	Permintaan di mana Amazon Cognito tidak mengidentifikasi risiko apa pun.	Operasi: Jenis operasi, seperti PasswordChange, SignIn, atau SignUp.  UserPoolId: Pengidentifikasi kumpulan pengguna.

Amazon Cognito menawarkan dua grup metrik yang telah ditentukan untuk analisis siap pakai. CloudWatch Berdasarkan Klasifikasi Risiko mengidentifikasi perincian tingkat risiko untuk permintaan yang diidentifikasi oleh Amazon Cognito sebagai berisiko. Berdasarkan Klasifikasi Permintaan mencerminkan metrik yang dikumpulkan berdasarkan tingkat permintaan.

Grup Metrik Agregasi	Deskripsi
Berdasarkan Klasifikasi Risiko	Permintaan yang diidentifikasi Amazon Cognito sebagai berisiko.
Berdasarkan Klasifikasi Permintaan	Metrik yang digabungkan berdasarkan permintaan.

## Mengaktifkan keamanan lanjutan kumpulan pengguna dari aplikasi Anda

Setelah Anda mengonfigurasi fitur keamanan lanjutan untuk kumpulan pengguna Anda, Anda harus mengaktifkannya di web atau aplikasi seluler Anda.

## Menggunakan keamanan tingkat lanjut dengan JavaScript

1. Tambahkan [SDK Identitas Amazon Cognito JavaScript](#) ke aplikasi Anda.
2. Di [CognitoUserPool.js](#), atur `AdvancedSecurityDataCollectionFlag` ke `true`. Setel `UserPoolId` ke ID kumpulan pengguna Anda.
3. Tambahkan referensi sumber ini ke JavaScript file aplikasi Anda. Ganti `<region>` dengan Wilayah AWS dari daftar berikut: `us-east-1,us-east-2,us-west-2,eu-west-1,eu-west-2,ataueu-central-1`.

```
<script src="https://amazon-cognito-assets.<region>.amazoncognito.com/amazon-cognito-advanced-security-data.min.js"></script>
```

## Menggunakan keamanan tingkat lanjut dengan Android

1. Buat aplikasi Anda dengan AWS Amplify untuk Android. Untuk informasi selengkapnya, lihat [Pengaturan Proyek](#) di Pusat AWS Amplify Pengembang.
2. Dengan `userContextDataProvider`, sertakan informasi pengguna dan perangkat dalam permintaan otentikasi Anda.

Untuk informasi tentang menambahkan data konteks pengguna di [Android SDK lama](#), lihat [aws-android-sdk - identityprovider-asf](#).cognito

## Menggunakan keamanan tingkat lanjut dengan iOS

1. Buat aplikasi Anda dengan AWS Amplify Swift atau Flutter. Untuk informasi selengkapnya, lihat Swift [Project Setup](#) dan Flutter [Project Setup](#) di AWS Amplify Dev Center.
2. Sertakan informasi pengguna dan perangkat dalam permintaan otentikasi Anda. Untuk contoh yang akan digunakan dengan operasi [InitiateAuth](#) API, lihat `userContextData` di [InitiateAuthInput+Amplify.Swift](#) on. GitHub

Untuk informasi tentang menambahkan data konteks pengguna di [SDK iOS lama](#), lihat [AWSCognitoIdentityProviderASF](#)

## Mengaitkan ACL AWS WAF web dengan kumpulan pengguna

AWS WAF adalah firewall aplikasi web. Dengan daftar kontrol akses AWS WAF web (web ACL), Anda dapat melindungi kumpulan pengguna dari permintaan yang tidak diinginkan ke UI yang

dihosting dan titik akhir layanan Amazon Cognito API. ACL web memberi Anda kontrol halus atas semua permintaan web HTTPS yang ditanggapi oleh kumpulan pengguna Anda. Untuk informasi selengkapnya tentang ACL AWS WAF web, lihat [Mengelola dan menggunakan daftar kontrol akses web \(web ACL\) di Panduan AWS WAF](#) Pengembang.

Jika Anda memiliki ACL AWS WAF web yang terkait dengan kumpulan pengguna, Amazon Cognito meneruskan header dan konten permintaan non-rahasia yang dipilih dari pengguna Anda. AWS WAF memeriksa isi permintaan, membandingkannya dengan aturan yang Anda tentukan di ACL web Anda, dan mengembalikan respons ke Amazon Cognito.

## Hal-hal yang perlu diketahui tentang ACL AWS WAF web dan Amazon Cognito

- Permintaan yang diblokir AWS WAF tidak dihitung terhadap kuota tingkat permintaan untuk jenis permintaan apa pun. AWS WAF Handler dipanggil sebelum penanganan pelambatan tingkat API.
- Saat Anda membuat ACL web, sejumlah kecil waktu berlalu sebelum ACL web sepenuhnya disebar dan tersedia untuk Amazon Cognito. Waktu propagasi bisa dari beberapa detik hingga beberapa menit. AWS WAF mengembalikan a [WAFUnavailableEntityException](#) ketika Anda mencoba untuk mengaitkan ACL web sebelum sepenuhnya disebar.
- Anda dapat mengaitkan satu ACL web dengan kumpulan pengguna.
- Permintaan Anda mungkin menghasilkan muatan yang lebih besar dari batas apa yang AWS WAF dapat diperiksa. Lihat [Penanganan komponen permintaan ukuran besar](#) di Panduan AWS WAF Pengembang untuk mempelajari cara mengonfigurasi cara AWS WAF menangani permintaan oversized dari Amazon Cognito.
- Anda tidak dapat mengaitkan ACL web yang menggunakan [pencegahan pengambilalihan akun Kontrol AWS WAF Penipuan \(ATP\)](#) dengan kumpulan pengguna Amazon Cognito. Anda menerapkan fitur ATP saat menambahkan grup aturan AWS-`AWSMANAGEDRULES`ATPRuleSet dikelola. Sebelum Anda mengaitkannya dengan kumpulan pengguna, pastikan ACL web Anda tidak menggunakan grup aturan dikelola ini.
- Bila Anda memiliki ACL AWS WAF web yang terkait dengan kumpulan pengguna, dan aturan di ACL web Anda menyajikan CAPTCHA, ini dapat menyebabkan kesalahan yang tidak dapat dipulihkan dalam pendaftaran TOTP UI yang dihosting. Untuk membuat aturan yang memiliki tindakan CAPTCHA dan tidak memengaruhi TOTP UI yang dihosting, lihat. [Mengonfigurasi ACL AWS WAF web Anda untuk MFA TOTP UI yang dihosting](#)

AWS WAF memeriksa permintaan ke titik akhir berikut.

## UI yang Dihosting

Permintaan ke semua titik akhir di [Titik akhir federasi kumpulan pengguna dan referensi UI yang dihosting](#)

### Operasi API publik

Permintaan dari aplikasi Anda ke Amazon Cognito API yang tidak menggunakan AWS kredensial untuk mengotorisasi. Ini termasuk operasi API seperti [InitiateAuth](#), [RespondToAuthChallenge](#), dan [GetUser](#). Operasi API yang berada dalam lingkup AWS WAF tidak memerlukan otentikasi dengan AWS kredensial. Mereka tidak diautentikasi, atau diotorisasi dengan string sesi atau token akses. Untuk informasi selengkapnya, lihat [Kumpulan pengguna Amazon Cognito operasi API yang diautentikasi dan tidak diautentikasi](#).

Anda dapat mengonfigurasi aturan di ACL web Anda dengan tindakan aturan yang Menghitung, Mengizinkan, Memblokir, atau menampilkan CAPTCHA sebagai respons terhadap permintaan yang cocok dengan aturan. Untuk informasi selengkapnya, lihat [AWS WAF aturan](#) di Panduan AWS WAF Pengembang. Bergantung pada tindakan aturan, Anda dapat menyesuaikan respons yang dikembalikan Amazon Cognito ke pengguna Anda.

#### Important

Opsi Anda untuk menyesuaikan respons kesalahan bergantung pada cara Anda membuat permintaan API.

- Anda dapat menyesuaikan kode kesalahan dan badan respons permintaan UI yang dihosting. Anda hanya dapat menyajikan CAPTCHA untuk dipecahkan oleh pengguna Anda di UI yang dihosting.
- Untuk permintaan yang Anda buat dengan [API kumpulan pengguna](#) Amazon Cognito, Anda dapat menyesuaikan badan respons permintaan yang menerima respons Blokir. Anda juga dapat menentukan kode kesalahan khusus dalam kisaran 400-499.
- The AWS Command Line Interface (AWS CLI) dan AWS SDK mengembalikan `ForbiddenException` kesalahan ke permintaan yang menghasilkan respons Block atau CAPTCHA.

## Mengaitkan ACL web dengan kumpulan pengguna Anda

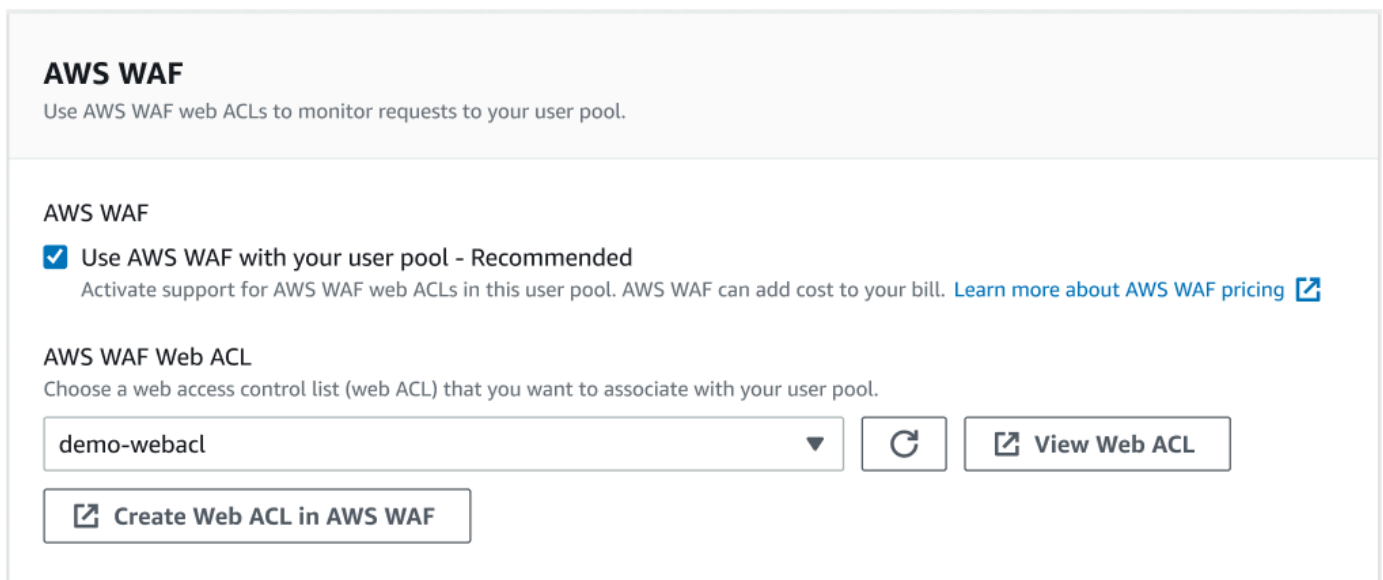
Untuk bekerja dengan ACL web di kumpulan pengguna Anda, prinsipal AWS Identity and Access Management (IAM) Anda harus memiliki izin Amazon Cognito berikut. Untuk informasi tentang AWS WAF izin, lihat [izin AWS WAF API di Panduan AWS WAF](#) Pengembang.

- `cognito-idp:AssociateWebACL`
- `cognito-idp:DisassociateWebACL`
- `cognito-idp:GetWebACLForResource`
- `cognito-idp:ListResourcesForWebACL`

[Meskipun Anda harus memberikan izin IAM, tindakan yang tercantum hanya izin dan tidak sesuai dengan operasi API.](#)

AWS WAF Untuk mengaktifkan kumpulan pengguna Anda dan mengaitkan ACL web

1. Masuk ke [konsol Amazon Cognito](#).
2. Di panel navigasi, pilih Kumpulan Pengguna, dan pilih kumpulan pengguna yang ingin Anda edit.
3. Pilih tab Properti kumpulan pengguna.
4. Pilih Edit di sebelah AWS WAF.
5. Di bawah AWS WAF, pilih Gunakan AWS WAF dengan kumpulan pengguna Anda.






**AWS WAF**  
Use AWS WAF web ACLs to monitor requests to your user pool.

**AWS WAF**

Use AWS WAF with your user pool - Recommended  
Activate support for AWS WAF web ACLs in this user pool. AWS WAF can add cost to your bill. [Learn more about AWS WAF pricing](#)

**AWS WAF Web ACL**  
Choose a web access control list (web ACL) that you want to associate with your user pool.

demo-webacl ▼  



6. Pilih ACL AWS WAF Web yang sudah Anda buat, atau pilih Buat ACL web AWS WAF untuk membuatnya di AWS WAF sesi baru di. AWS Management Console

## 7. Pilih Simpan perubahan.

Untuk mengaitkan ACL web secara terprogram dengan kumpulan pengguna Anda di AWS Command Line Interface atau SDK, gunakan [AssociateWebACL](#) dari API. AWS WAF Amazon Cognito tidak memiliki operasi API terpisah yang mengaitkan ACL web.

### Menguji dan mencatat ACL AWS WAF web

Saat Anda menetapkan tindakan aturan ke Count di ACL web Anda, AWS WAF tambahkan permintaan ke hitungan permintaan yang cocok dengan aturan. Untuk menguji ACL web dengan kumpulan pengguna Anda, tetapkan tindakan aturan ke Hitung dan pertimbangkan volume permintaan yang cocok dengan setiap aturan. Misalnya, jika aturan yang ingin disetel ke tindakan Blokir cocok dengan sejumlah besar permintaan yang Anda tentukan sebagai lalu lintas pengguna normal, Anda mungkin perlu mengonfigurasi ulang aturan Anda. Untuk informasi selengkapnya, lihat [Menguji dan menyetel AWS WAF perlindungan Anda](#) di Panduan AWS WAF Pengembang.

Anda juga dapat mengonfigurasi header permintaan log AWS WAF ke grup CloudWatch log Amazon Logs, bucket Amazon Simple Storage Service (Amazon S3), atau Amazon Data Firehose. Anda dapat mengidentifikasi permintaan Amazon Cognito yang Anda buat dengan API kumpulan pengguna oleh `x-amzn-cognito-client-id`. `x-amzn-cognito-operation-name` Permintaan UI yang di-host hanya menyertakan `x-amzn-cognito-client-id` header. Untuk informasi selengkapnya, lihat [Mencatat lalu lintas ACL web](#) di Panduan AWS WAF Pengembang.

AWS WAF ACL web tidak tunduk pada [harga untuk fitur keamanan lanjutan Amazon Cognito](#). Fitur keamanan AWS WAF melengkapi fitur keamanan canggih Amazon Cognito. Anda dapat mengaktifkan kedua fitur di kumpulan pengguna. AWS WAF tagihan secara terpisah untuk pemeriksaan permintaan kumpulan pengguna. Untuk informasi selengkapnya, silakan lihat [Harga AWS WAF](#).

Data AWS WAF permintaan logging tunduk pada penagihan tambahan oleh layanan tempat Anda menargetkan log Anda. Untuk informasi selengkapnya, lihat [Harga untuk mencatat informasi lalu lintas ACL web](#) di Panduan AWS WAF Pengembang.

### Sensitivitas casing kumpulan pengguna

Kumpulan pengguna Amazon Cognito yang Anda buat secara default tidak peka huruf besar/kecil. AWS Management Console Jika kumpulan pengguna tidak peka huruf besar/kecil, `user@example.com` dan `User@example.com` merujuk ke pengguna yang sama. Ketika



nama pengguna dalam kumpulan pengguna tidak peka huruf besar/kecil, email atribut `preferred_username` and juga tidak peka huruf besar/kecil.

Untuk memperhitungkan setelan sensitivitas huruf besar kumpulan pengguna, identifikasi pengguna dalam kode aplikasi Anda berdasarkan atribut pengguna alternatif. Karena kasus nama pengguna, nama pengguna yang disukai, atau atribut alamat email dapat bervariasi di profil pengguna yang berbeda, rujuk ke sub atribut. Anda juga dapat membuat atribut kustom yang tidak dapat diubah di kumpulan pengguna, dan menetapkan nilai pengenalan unik Anda sendiri ke atribut di setiap profil pengguna baru. Saat pertama kali membuat pengguna, Anda dapat menulis nilai ke atribut kustom yang tidak dapat diubah yang Anda buat.

#### Note

Terlepas dari pengaturan sensitivitas kasus pada kumpulan pengguna Anda, Amazon Cognito mengharuskan pengguna gabungan dari penyedia identitas SAMP atau OIDC (iDP) meneruskan klaim atau unik dan peka huruf besar/kecil. NameId sub Untuk informasi selengkapnya tentang sensitivitas kasus pengenalan unik dan SAMP IdPs, lihat. [Menggunakan login SAMP yang diinisialisasi SP](#)

## Membuat kumpulan pengguna yang peka huruf besar/kecil

Jika Anda membuat resource dengan operasi AWS Command Line Interface (AWS CLI) dan API seperti [CreateUserPool](#), Anda harus menyetel `CaseSensitive` parameter Boolean ke `false`. Pengaturan ini membuat kumpulan pengguna yang tidak peka huruf besar/kecil. Jika Anda tidak menentukan nilai, `CaseSensitive` defaultnya `true`. Default ini adalah kebalikan dari perilaku default untuk kumpulan pengguna yang Anda buat di AWS Management Console. Sebelum 12 Februari 2020, kumpulan pengguna default menjadi peka huruf besar/kecil terlepas dari platformnya.

Anda dapat menggunakan tab Pengalaman masuk AWS Management Console atau operasi [DescribeUserPool](#) API untuk meninjau setelan sensitivitas kasus untuk setiap kumpulan pengguna di akun Anda.

## Migrasi ke kumpulan pengguna baru

Karena potensi konflik antar profil pengguna, Anda tidak dapat mengubah kumpulan pengguna Amazon Cognito dari peka huruf besar/kecil menjadi tidak peka huruf besar/kecil. Sebagai

gantinya, memigrasikan pengguna Anda ke kumpulan pengguna baru. Anda harus membuat kode migrasi untuk menyelesaikan konflik terkait kasus. Kode ini harus mengembalikan pengguna baru yang unik atau menolak upaya masuk saat mendeteksi konflik. Di kumpulan pengguna baru yang tidak peka huruf besar/kecil, tetapkan file. [Memigrasi pengguna pemicu Lambda](#) AWS Lambda Fungsi ini dapat membuat pengguna di kumpulan pengguna baru yang tidak peka huruf besar/kecil. Ketika pengguna gagal masuk dengan kumpulan pengguna yang tidak peka huruf besar/kecil, fungsi Lambda menemukan dan menduplikasi pengguna dari kumpulan pengguna yang peka huruf besar/kecil. Anda juga dapat mengaktifkan pemicu [ForgotPassword](#) Lambda pengguna yang bermigrasi pada acara. Amazon Cognito meneruskan informasi pengguna dan metadata peristiwa dari tindakan masuk atau pemulihan kata sandi ke fungsi Lambda Anda. Anda dapat menggunakan data peristiwa untuk mengelola konflik antara nama pengguna dan alamat email saat fungsi Anda membuat pengguna baru di kumpulan pengguna yang tidak peka huruf besar/kecil. Konflik ini antara nama pengguna dan alamat email yang unik di kumpulan pengguna yang tidak peka huruf besar/kecil, tetapi identik dalam kumpulan pengguna yang peka huruf besar/kecil.


Untuk informasi selengkapnya tentang cara menggunakan pemicu Lambda pengguna yang bermigrasi di antara kumpulan pengguna Amazon Cognito, [lihat Memigrasi Pengguna ke kumpulan pengguna Amazon Cognito](#) di blog. AWS

## perlindungan penghapusan kolam pengguna

Untuk membuatnya agar administrator Anda tidak sengaja menghapus pangkalan pengguna Anda, aktifkan perlindungan penghapusan. Dengan perlindungan penghapusan aktif, Anda harus mengonfirmasi bahwa Anda ingin menghapus pangkalan pengguna sebelum menghapusnya. Ketika Anda menghapus pangkalan pengguna di AWS Management Console, Anda dapat menonaktifkan perlindungan penghapusan pada saat yang sama. Saat Anda menerima permintaan untuk menonaktifkan perlindungan penghapusan dan mengonfirmasi niat Anda untuk menghapus, seperti yang ditunjukkan pada gambar berikut, Amazon Cognito menghapus kumpulan pengguna Anda.

## Delete user pool [redacted] ? ✕

Before you delete this user pool, first make sure no services or apps rely on it.

 If you delete this user pool, and your app still relies on it, any sign-in and sign-up attempts will fail.

- To delete this user pool, permit Amazon Cognito to also take the following prerequisite actions.
  - Deactivate deletion protection**
- To confirm deletion, enter `testUserPool` in the field.

Cancel Delete

Ketika Anda ingin menghapus pangkalan pengguna dengan permintaan Amazon Cognito API, Anda harus terlebih dahulu mengubah `DeletionProtection` ke `InActive` dalam [UpdateUserPool](#) permintaan. Jika Anda tidak menonaktifkan perlindungan penghapusan, Amazon Cognito mengembalikan `InvalidParameterException` kesalahan. Setelah menonaktifkan perlindungan penghapusan, Anda dapat menghapus pangkalan pengguna dalam [DeleteUserPool](#) permintaan.

Amazon Cognito mengaktifkan perlindungan Penghapusan secara default saat Anda membuat pangkalan pengguna baru di AWS Management Console. Saat Anda membuat pangkalan pengguna dengan `CreateUserPool` API, perlindungan penghapusan tidak aktif secara default. Untuk menggunakan fitur ini di kumpulan pengguna yang Anda buat dengan AWS CLI atau AWS SDK, atur `DeletionProtection` parameternya `True`.

Anda dapat mengaktifkan atau menonaktifkan status perlindungan penghapusan di wadah perlindungan Penghapusan di tab Pengaturan pangkalan pengguna di konsol Amazon Cognito.

Cara mengkonfigurasi perlindungan penghapusan

- Masuk ke [Konsol Amazon Cognito](#). Anda mungkin diminta untuk kredensial AWS Anda.
- Pilih User Pools.

3. Pilih kolom pengguna yang ada dari daftar, atau [buat kolom pengguna](#).
4. Pilih tab Pengaturan pangkalan pengguna. Temukan Perlindungan Penghapusan dan pilih Aktifkan atau Nonaktifkan.
5. Konfirmasikan pilihan Anda dalam dialog berikutnya.

## Mengelola respons kesalahan keberadaan pengguna

Amazon Cognito mendukung penyesuaian respons kesalahan yang dikembalikan oleh kumpulan pengguna. Respons kesalahan khusus tersedia untuk pembuatan dan otentikasi pengguna, pemulihan kata sandi, dan operasi konfirmasi.

Gunakan pengaturan `PreventUserExistenceErrors` dari klien aplikasi kolam pengguna untuk mengaktifkan atau menonaktifkan keberadaan pengguna terkait kesalahan. Saat Anda membuat klien aplikasi baru dengan API kumpulan pengguna Amazon Cognito, akan, `PreventUserExistenceErrors` atau dinonaktifkan `LEGACY`, secara default. Di konsol Amazon Cognito, opsi Cegah kesalahan keberadaan pengguna —setelan `ENABLED` untuk `PreventUserExistenceErrors` —dipilih secara default. Untuk memperbarui `PreventUserExistenceErrors` konfigurasi Anda, lakukan salah satu hal berikut:

- Ubah nilai `PreventUserExistenceErrors` antara `ENABLED` dan `LEGACY` dalam permintaan [UpdateUserPoolClientAPI](#).
- Edit klien aplikasi Anda di konsol Amazon Cognito dan ubah status Cegah kesalahan keberadaan pengguna antara `selected` (`ENABLED`) dan `deselected` (`LEGACY`).

Jika properti ini memiliki nilai `LEGACY`, klien aplikasi Anda akan menampilkan respons `UserNotFoundException` kesalahan saat pengguna mencoba masuk dengan nama pengguna yang tidak ada di kumpulan pengguna Anda.

Jika properti ini memiliki nilai `ENABLED`, klien aplikasi Anda tidak mengungkapkan tidak adanya akun pengguna di kumpulan pengguna Anda dengan `UserNotFoundException` kesalahan. `PreventUserExistenceErrors` konfigurasi `ENABLED` memiliki efek sebagai berikut:

- Amazon Cognito merespons dengan informasi nonspesifik terhadap permintaan API di mana tanggapannya mungkin mengungkapkan bahwa ada pengguna yang valid.
- API login dan lupa kata sandi Amazon Cognito mengembalikan respons kegagalan autentikasi umum. Respons kesalahan memberitahu Anda nama pengguna atau kata sandi salah.

- Konfirmasi akun Amazon Cognito dan API pemulihan kata sandi mengembalikan respons yang menunjukkan kode dikirim ke media pengiriman simulasi, bukan representasi sebagian dari informasi kontak pengguna.

Informasi berikut merinci perilaku operasi kumpulan pengguna saat `PreventUserExistenceErrors` disetel ke `ENABLED`.

## Operasi otentikasi dan pembuatan pengguna

Anda dapat mengonfigurasi respons kesalahan dalam autentikasi Username-Password dan Secure Remote Password (SRP). Anda juga dapat menyesuaikan kesalahan yang Anda kembalikan dengan otentikasi khusus. API berikut melakukan operasi autentikasi ini:

- `AdminInitiateAuth`
- `AdminRespondToAuthChallenge`
- `InitiateAuth`
- `RespondToAuthChallenge`

Daftar berikut menunjukkan bagaimana Anda dapat menyesuaikan respons kesalahan dalam operasi otentikasi pengguna.

### Otentikasi nama pengguna dan kata sandi

Untuk menandatangani pengguna dengan `ADMIN_USER_PASSWORD_AUTH` dan `USER_PASSWORD_AUTH`, sertakan nama pengguna dan kata sandi dalam permintaan `AdminInitiateAuth` atau `InitiateAuth` API. Amazon Cognito mengembalikan `NotAuthorizedException` kesalahan umum ketika nama pengguna atau kata sandi salah.

### Autentikasi berbasis Secure Remote Password (SRP)

Untuk menandatangani pengguna `USER_SRP_AUTH`, sertakan nama pengguna dan `SRP_A` parameter dalam permintaan `AdminInitiateAuth` atau `InitiateAuth` API. Sebagai tanggapan, Amazon Cognito kembali `SRP_B` dan garam untuk pengguna. Ketika pengguna tidak ditemukan, Amazon Cognito mengembalikan respons yang disimulasikan pada langkah pertama seperti yang dijelaskan di [RFC 5054](#). Amazon Cognito mengembalikan garam yang sama dan ID pengguna internal dalam format [Universally Unique Identifier \(UUID\)](#) untuk kombinasi nama pengguna dan kumpulan pengguna yang sama. Saat Anda mengirim permintaan

RespondToAuthChallenge API dengan bukti kata sandi, Amazon Cognito mengembalikan `NotAuthorizedException` kesalahan umum ketika nama pengguna atau kata sandi salah.

#### Note

Anda dapat mensimulasikan respons generik dengan autentikasi nama pengguna dan kata sandi jika Anda menggunakan atribut alias berbasis verifikasi, dan nama pengguna yang tidak dapat diubah tidak diformat sebagai UUID.

## Tantangan Autentikasi Kustom pemacu Lambda

Jika Anda menggunakan Pemacu [Lambda Tantangan Otentikasi Kustom](#) dan Anda mengaktifkan respons kesalahan, `LambdaChallenge` maka mengembalikan parameter Boolean bernama `UserNotFound` Kemudian diteruskan dalam permintaan `DefineAuthChallenge`, `VerifyAuthChallenge`, dan pemacu Lambda `CreateAuthChallenge`. Anda dapat menggunakan pemacu ini untuk mensimulasikan tantangan otorisasi kustom untuk pengguna yang tidak ada. Jika Anda memanggil pemacu Pra-Autentikasi Lambda untuk pengguna yang tidak ada, maka Amazon Cognito mengembalikan `UserNotFound`.

Daftar berikut menunjukkan bagaimana Anda dapat menyesuaikan respons kesalahan dalam operasi pembuatan pengguna.

### SignUp

`SignUpOperasi` selalu kembali `UsernameExistsException` ketika nama pengguna sudah diambil. Jika Anda tidak ingin Amazon Cognito menampilkan `UsernameExistsException` kesalahan untuk alamat email dan nomor telepon saat mendaftar pengguna di aplikasi, gunakan atribut alias berbasis verifikasi. Untuk informasi selengkapnya tentang alias, lihat [Menyesuaikan atribut masuk](#).

Untuk contoh bagaimana Amazon Cognito dapat mencegah penggunaan permintaan `SignUp` API untuk menemukan pengguna di kumpulan pengguna Anda, lihat [Mencegah UsernameExistsException kesalahan untuk alamat email dan nomor telepon saat mendaftar](#)

### Mengimpor pengguna

Jika `PreventUserExistenceErrors` diaktifkan, selama otentikasi pengguna yang diimpor, `NotAuthorizedException` kesalahan umum dikembalikan yang menunjukkan nama pengguna atau kata sandi salah alih-alih kembali. `PasswordResetRequiredException`

Lihat [Membutuhkan pengguna yang diimpor untuk mengatur ulang kata sandi](#) untuk informasi selengkapnya.

## Memigrasi pengguna pemicu Lambda

Amazon Cognito mengembalikan respons yang disimulasikan untuk pengguna yang tidak ada ketika respons kosong diatur dalam konteks peristiwa asli oleh pemicu Lambda. Untuk informasi selengkapnya, lihat [Migrasi Pengguna Pemicu Lambda](#).

## Mencegah `UsernameExistsException` kesalahan untuk alamat email dan nomor telepon saat mendaftar

Contoh berikut menunjukkan bagaimana, ketika Anda mengonfigurasi atribut alias di kumpulan pengguna, Anda dapat menyimpan duplikat alamat email dan nomor telepon agar tidak menghasilkan `UsernameExistsException` kesalahan dalam menanggapi `SignUp` permintaan API. Anda harus telah membuat kumpulan pengguna Anda dengan alamat email atau nomor telepon sebagai atribut alias. Untuk informasi selengkapnya, lihat bagian [Menyesuaikan atribut masuk dari atribut kumpulan pengguna](#).

1. Jie mendaftar untuk nama pengguna baru, dan juga memberikan alamat `jie@example.com` email. Amazon Cognito mengirimkan kode ke alamat email mereka.

### Contoh AWS CLI perintah

```
aws cognito-idp sign-up --client-id 1234567890abcdef0 --username jie --password  
PASSWORD --user-attributes Name="email",Value="jie@example.com"
```

### Contoh respon

```
{  
  "UserConfirmed": false,  
  "UserSub": "<subId>",  
  "CodeDeliveryDetails": {  
    "AttributeName": "email",  
    "Destination": "j****@e****",  
    "DeliveryMedium": "EMAIL"  
  }  
}
```

2. Jie memberikan kode yang dikirim kepada mereka untuk mengonfirmasi kepemilikan mereka atas alamat email tersebut. Ini melengkapi pendaftaran mereka sebagai pengguna.

## Contoh AWS CLI perintah

```
aws cognito-idp confirm-sign-up --client-id 1234567890abcdef0 --username=jie --confirmation-code xxxxxx
```

3. Shirley mendaftarkan akun pengguna baru dan memberikan alamat email. `jie@example.com` Amazon Cognito tidak mengembalikan `UsernameExistsException` kesalahan, dan mengirimkan kode konfirmasi ke alamat email Jie.

## Contoh AWS CLI perintah

```
aws cognito-idp sign-up --client-id 1234567890abcdef0 --username shirley --password PASSWORD --user-attributes Name="email",Value="jie@example.com"
```

## Contoh respon

```
{
  "UserConfirmed": false,
  "UserSub": "<new subId>",
  "CodeDeliveryDetails": {
    "AttributeName": "email",
    "Destination": "j****@e****",
    "DeliveryMedium": "EMAIL"
  }
}
```

4. Dalam skenario yang berbeda, Shirley memiliki kepemilikan. `jie@example.com` Shirley mengambil kode yang dikirim Amazon Cognito ke alamat email Jie dan mencoba mengonfirmasi akun tersebut.

## Contoh AWS CLI perintah

```
aws cognito-idp confirm-sign-up --client-id 1234567890abcdef0 --username=shirley --confirmation-code xxxxxx
```

## Contoh respon

```
An error occurred (AliasExistsException) when calling the ConfirmSignUp operation: An account with the email already exists.
```



Amazon Cognito tidak mengembalikan kesalahan pada `aws cognito-idp sign-up` permintaan Shirley, meskipun telah ditetapkan ke pengguna `jie@example.com` yang sudah ada. Shirley harus menunjukkan kepemilikan alamat email sebelum Amazon Cognito mengembalikan respons kesalahan. Dalam kumpulan pengguna dengan atribut alias, perilaku ini mencegah penggunaan `SignUp` API publik untuk memeriksa apakah ada pengguna dengan alamat email atau nomor telepon tertentu.

Perilaku ini berbeda dengan respons yang dikembalikan Amazon Cognito ke `SignUp` permintaan dengan nama pengguna yang ada, seperti yang ditunjukkan pada contoh berikut. Sementara Shirley belajar dari tanggapan ini bahwa pengguna sudah ada dengan nama pengguna `jie`, mereka tidak belajar tentang alamat email atau nomor telepon yang terkait dengan pengguna.

### Contoh perintah CLI

```
aws cognito-idp sign-up --client-id lexample23456789 --username jie --password PASSWORD
--user-attributes Name="email",Value="shirley@example.com"
```

### Contoh respon

```
An error occurred (UsernameExistsException) when calling the SignUp operation: User
already exists
```

## Operasi reset kata sandi

Amazon Cognito mengembalikan respons berikut ke operasi pengaturan ulang kata sandi pengguna saat Anda mencegah kesalahan keberadaan pengguna.

### ForgotPassword

Ketika pengguna tidak ditemukan, dinonaktifkan, atau tidak memiliki mekanisme pengiriman terverifikasi untuk memulihkan kata sandi mereka, Amazon Cognito `CodeDeliveryDetails` kembali dengan media pengiriman simulasi untuk pengguna. Media pengiriman simulasi ditentukan oleh format nama pengguna input dan pengaturan verifikasi kumpulan pengguna.

### ConfirmForgotPassword

Amazon Cognito mengembalikan kesalahan `CodeMismatchException` untuk pengguna yang tidak ada atau dinonaktifkan. Jika kode tidak diminta saat menggunakan `ForgotPassword`, Amazon Cognito mengembalikan kesalahan `ExpiredCodeException`.

## Operasi konfirmasi

Amazon Cognito mengembalikan tanggapan berikut untuk konfirmasi pengguna dan operasi verifikasi saat Anda mencegah kesalahan keberadaan pengguna.

### ResendConfirmationCode

Amazon Cognito mengembalikan kesalahan `CodeDeliveryDetails` untuk pengguna yang dinonaktifkan atau pengguna yang tidak ada. Amazon Cognito mengirimkan kode konfirmasi ke email atau nomor telepon pengguna yang sudah ada.

### ConfirmSignUp

`ExpiredCodeException` akan kembali jika kode telah kedaluwarsa. Amazon Cognito mengembalikan `NotAuthorizedException` saat pengguna tidak diotorisasi. Jika kode tidak sesuai dengan apa yang diharapkan server, Amazon Cognito mengembalikan `CodeMismatchException`.

# Kumpulan identitas Amazon Cognito

Kumpulan identitas Amazon Cognito adalah direktori identitas federasi yang dapat Anda tukarkan dengan kredensialnya. AWS Kumpulan identitas menghasilkan AWS kredensial sementara untuk pengguna aplikasi Anda, apakah mereka telah masuk atau Anda belum mengidentifikasinya. Dengan peran dan kebijakan AWS Identity and Access Management (IAM), Anda dapat memilih tingkat izin yang ingin Anda berikan kepada pengguna Anda. Pengguna dapat memulai sebagai tamu dan mengambil aset yang Anda simpan. Layanan AWS Kemudian mereka dapat masuk dengan penyedia identitas pihak ketiga untuk membuka akses ke aset yang Anda sediakan untuk anggota terdaftar. Penyedia identitas pihak ketiga dapat berupa penyedia OAuth 2.0 konsumen (sosial) seperti Apple atau Google, penyedia identitas SAMP atau OIDC kustom, atau skema otentikasi khusus, juga disebut penyedia pengembang, dari desain Anda sendiri.

Fitur kumpulan identitas Amazon Cognito

Menandatangani permintaan untuk Layanan AWS

[Tanda tangani permintaan API](#) untuk Layanan AWS menyukai Amazon Simple Storage Service (Amazon S3) dan Amazon DynamoDB. Analisis aktivitas pengguna dengan layanan seperti Amazon Pinpoint dan Amazon CloudWatch

Filter permintaan dengan kebijakan berbasis sumber daya

Lakukan kontrol granular atas akses pengguna ke sumber daya Anda. Ubah klaim pengguna menjadi [tag sesi IAM](#), dan buat kebijakan IAM yang memberikan akses sumber daya ke subset pengguna Anda yang berbeda.

Tetapkan akses tamu

Untuk pengguna yang belum masuk, konfigurasi kumpulan identitas Anda untuk menghasilkan AWS kredensial dengan cakupan akses yang sempit. Mengotentikasi pengguna melalui penyedia masuk tunggal untuk meningkatkan akses mereka.

Tetapkan peran IAM berdasarkan karakteristik pengguna

Tetapkan peran IAM tunggal ke semua pengguna yang diautentikasi, atau pilih peran berdasarkan klaim setiap pengguna.

Terima berbagai penyedia identitas

Tukarkan ID atau token akses, token kumpulan pengguna, pernyataan SAMP, atau token OAuth penyedia sosial untuk kredensial AWS

## Validasi identitas Anda sendiri

Lakukan validasi pengguna Anda sendiri dan gunakan kredensi pengembang Anda untuk mengeluarkan AWS kredensi bagi pengguna Anda.

Anda mungkin sudah memiliki kumpulan pengguna Amazon Cognito yang menyediakan layanan autentikasi dan otorisasi ke aplikasi Anda. Anda dapat mengatur kumpulan pengguna sebagai penyedia identitas (iDP) ke kumpulan identitas Anda. Ketika Anda melakukannya, pengguna Anda dapat mengautentikasi melalui kumpulan pengguna Anda IdPs, mengkonsolidasikan klaim mereka ke dalam token identitas OIDC umum, dan menukar token tersebut dengan kredensial. AWS Pengguna Anda kemudian dapat menunjukkan kredensialnya dalam permintaan yang ditandatangani kepada Anda. Layanan AWS

Anda juga dapat mengajukan klaim yang diautentikasi dari salah satu penyedia identitas Anda langsung ke kumpulan identitas Anda. Amazon Cognito menyesuaikan klaim pengguna dari penyedia SAMP, OAuth, dan OIDC menjadi permintaan API untuk kredensi jangka pendek.

[AssumeRoleWithWebIdentity](#)

Kumpulan pengguna Amazon Cognito seperti penyedia identitas OIDC ke aplikasi berkemampuan SSO Anda. Kumpulan identitas bertindak sebagai penyedia AWSidentitas untuk aplikasi apa pun dengan dependensi sumber daya yang paling sesuai dengan otorisasi IAM.

Kolam identitas Amazon Cognito mendukung penyedia identitas berikut:

- Penyedia publik: [Menyiapkan Login with Amazon sebagai kumpulan identitas IDP](#), [Menyiapkan Facebook sebagai kumpulan identitas IDP](#), [Menyiapkan Google sebagai kumpulan identitas iDP](#), [Menyiapkan Masuk dengan Apple sebagai kumpulan identitas iDP](#), Twitter.
- [Kumpulan pengguna Amazon Cognito](#)
- [Menyiapkan penyedia OIDC sebagai idP kumpulan identitas](#)
- [Menyiapkan penyedia SAMP sebagai idP kumpulan identitas](#)
- [Identitas yang diautentikasi pengembang \(kumpulan identitas\)](#)

Untuk informasi tentang ketersediaan Wilayah kolam identitas Amazon Cognito, lihat [AWS Ketersediaan Wilayah Layanan](#).

Untuk informasi selengkapnya tentang kolam identitas Amazon Cognito, lihat topik-topik berikut.

Topik

- [Menggunakan kumpulan identitas \(identitas federasi\)](#)
- [Konsep kolam identitas](#)
- [Praktik terbaik keamanan untuk kumpulan identitas Amazon Cognito](#)
- [Menggunakan atribut untuk kontrol akses](#)
- [Menggunakan kontrol akses berbasis peran](#)
- [Mendapatkan kredensial](#)
- [Mengakses layanan AWS](#)
- [Identitas mengumpulkan penyedia identitas eksternal](#)
- [Identitas yang diautentikasi pengembang \(kumpulan identitas\)](#)
- [Mengalihkan pengguna yang tidak diautentikasi ke pengguna terautentikasi \(kumpulan identitas\)](#)

## Menggunakan kumpulan identitas (identitas federasi)

Kumpulan identitas Amazon Cognito menyediakan AWS kredensial sementara untuk pengguna yang merupakan tamu (tidak diautentikasi) dan untuk pengguna yang telah diautentikasi dan menerima token. Kolam identitas adalah penyimpanan data identitas pengguna yang spesifik untuk akun Anda.

Untuk membuat kolam identitas baru di konsol

1. Masuk ke [konsol Amazon Cognito](#) dan pilih Identity pool.
2. Pilih Buat kumpulan identitas.
3. Di Konfigurasi kepercayaan kumpulan identitas, pilih untuk menyiapkan kumpulan identitas Anda untuk akses Terautentikasi, akses Tamu, atau keduanya.
  - Jika Anda memilih Akses yang diautentikasi, pilih satu atau beberapa jenis Identitas yang ingin Anda tetapkan sebagai sumber identitas yang diautentikasi di kumpulan identitas Anda. Jika mengonfigurasi penyedia pengembang Kustom, Anda tidak dapat memodifikasi atau menghapusnya setelah membuat kumpulan identitas.
4. Di Konfigurasi izin, pilih peran IAM default untuk pengguna yang diautentikasi atau tamu di kumpulan identitas Anda.
  - a. Pilih untuk Membuat peran IAM baru jika Anda ingin Amazon Cognito membuat peran baru untuk Anda dengan izin dasar dan hubungan kepercayaan dengan kumpulan identitas Anda. Masukkan nama peran IAM untuk mengidentifikasi peran baru Anda,

- misalnya `identitypool_authenticatedrole`. Pilih Lihat dokumen kebijakan untuk meninjau izin yang akan ditetapkan Amazon Cognito ke peran IAM baru Anda.
- b. Anda dapat memilih untuk Menggunakan peran IAM yang ada jika Anda sudah memiliki peran Akun AWS yang ingin Anda gunakan. Anda harus mengonfigurasi kebijakan kepercayaan peran IAM Anda untuk disertakan `cognito-identity.amazonaws.com`. Konfigurasi kebijakan kepercayaan peran Anda agar hanya mengizinkan Amazon Cognito mengambil peran saat menampilkan bukti bahwa permintaan tersebut berasal dari pengguna yang diautentikasi di kumpulan identitas spesifik Anda. Untuk informasi selengkapnya, lihat [Kepercayaan peran dan izin](#).
5. Di Connect identity providers, masukkan detail penyedia identitas (IdPs) yang Anda pilih di Configure identity pool trust. Anda mungkin diminta untuk memberikan informasi klien aplikasi OAuth, memilih kumpulan pengguna Amazon Cognito, memilih IDP IAM, atau memasukkan pengenal khusus untuk penyedia pengembang.
- a. Pilih pengaturan Peran untuk setiap IDP. Anda dapat menetapkan pengguna dari IDP tersebut peran Default yang Anda atur saat mengonfigurasi peran Terautentikasi, atau Anda dapat Memilih peran dengan aturan. Dengan IdP kumpulan pengguna Amazon Cognito, Anda juga dapat Memilih peran dengan `preferred_role` dalam token. Untuk informasi lebih lanjut tentang `cognito:preferred_role` klaim, lihat [Menetapkan nilai prioritas ke grup](#).
    - i. Jika Anda memilih Pilih peran dengan aturan, masukkan Klaim sumber dari autentikasi pengguna Anda, Operator yang ingin Anda bandingkan dengan klaim, Nilai yang akan menyebabkan kecocokan dengan pilihan peran ini, dan Peran yang ingin Anda tetapkan saat penetapan Peran cocok. Pilih Tambahkan yang lain untuk membuat aturan tambahan berdasarkan kondisi yang berbeda.
    - ii. Pilih Resolusi Peran. Jika klaim pengguna tidak sesuai dengan aturan, Anda dapat menolak kredensi atau mengeluarkan kredensi untuk peran yang Diautentikasi.
  - b. Konfigurasi Atribut untuk kontrol akses untuk setiap IDP. Atribut untuk kontrol akses memetakan klaim pengguna ke [tag utama](#) yang diterapkan Amazon Cognito untuk sesi sementara mereka. Anda dapat membuat kebijakan IAM untuk memfilter akses pengguna berdasarkan tag yang Anda terapkan pada sesi mereka.
    - i. Untuk tidak menerapkan tag utama, pilih Tidak aktif.
    - ii. Untuk menerapkan tag utama berdasarkan sub dan aud klaim, pilih Gunakan pemetaan default.

- iii. Untuk membuat skema atribut kustom Anda sendiri ke tag utama, pilih Gunakan pemetaan khusus. Kemudian masukkan kunci Tag yang ingin Anda sumber dari setiap Klaim yang ingin Anda wakili dalam tag.
6. Di Konfigurasi properti, masukkan Nama di bawah nama kumpulan Identitas.
7. Di bawah Autentikasi dasar (klasik), pilih apakah Anda ingin Aktifkan aliran dasar. Dengan aliran dasar aktif, Anda dapat melewati pilihan peran yang dibuat untuk Anda IdPs dan menelepon [AssumeRoleWithWebIdentity](#) secara langsung. Untuk informasi selengkapnya, lihat [Aliran otentikasi kumpulan identitas \(identitas federasi\)](#).
8. Di bawah Tag, pilih Tambahkan tag jika Anda ingin menerapkan [tag](#) ke kumpulan identitas Anda.
9. Di Tinjau dan buat, konfirmasi pilihan yang Anda buat untuk kumpulan identitas baru Anda. Pilih Edit untuk kembali ke wizard dan mengubah pengaturan apa pun. Setelah selesai, pilih Buat kumpulan identitas.

## Peran IAM pengguna

Peran IAM menentukan izin bagi pengguna Anda untuk mengakses AWS sumber daya, seperti [Amazon Cognito Sync](#). Pengguna aplikasi Anda akan mengambil peran yang Anda buat. Anda dapat menentukan peran yang berbeda untuk pengguna yang terautentikasi dan tidak terautentikasi. Untuk mempelajari selengkapnya tentang IAM role, lihat [Peran IAM](#).

## Identitas yang diautentikasi dan tidak diautentikasi

Kolam identitas Amazon Cognito mendukung identitas terautentikasi dan tidak terautentikasi. Identitas terautentikasi adalah milik pengguna yang diautentikasi oleh penyedia identitas yang didukung. Identitas yang tidak diautentikasi umumnya adalah milik pengguna tamu.

- Untuk mengkonfigurasi identitas terautentikasi dengan penyedia login publik, lihat [Identitas mengumpulkan penyedia identitas eksternal](#).
- Untuk mengkonfigurasi proses autentikasi backend Anda sendiri, lihat [Identitas yang diautentikasi pengembang \(kumpulan identitas\)](#).

## Aktifkan atau nonaktifkan akses tamu

Identitas Amazon Cognito pool akses tamu (identitas yang tidak diautentikasi) menyediakan pengenal dan AWS kredensial unik untuk pengguna yang tidak mengautentikasi dengan penyedia identitas. Jika aplikasi Anda mengizinkan pengguna yang tidak masuk, Anda dapat mengaktifkan akses untuk

identitas yang tidak diautentikasi. Untuk mempelajari selengkapnya, lihat [Memulai dengan kumpulan identitas Amazon Cognito](#).

Untuk memperbarui akses tamu di kolam identitas

1. Pilih kumpulan Identitas dari konsol [Amazon Cognito](#). Pilih kumpulan identitas.
2. Pilih tab Akses pengguna.
3. Temukan akses Tamu. Di kumpulan identitas yang saat ini tidak mendukung akses tamu, Status Tidak Aktif.
  - a. Jika akses Tamu Aktif dan Anda ingin menonaktifkannya, pilih Nonaktifkan.
  - b. Jika akses Tamu Tidak Aktif dan Anda ingin mengaktifkannya, pilih Edit.
    - Pilih peran IAM default untuk pengguna tamu di kumpulan identitas Anda.
      - A. Pilih untuk Membuat peran IAM baru jika Anda ingin Amazon Cognito membuat peran baru untuk Anda dengan izin dasar dan hubungan kepercayaan dengan kumpulan identitas Anda. Masukkan nama peran IAM untuk mengidentifikasi peran baru Anda, misalnya `myidentitypool1_authenticatedrole`. Pilih Lihat dokumen kebijakan untuk meninjau izin yang akan ditetapkan Amazon Cognito ke peran IAM baru Anda.
      - B. Anda dapat memilih untuk Menggunakan peran IAM yang ada jika Anda sudah memiliki peran Akun AWS yang ingin Anda gunakan. Anda harus mengonfigurasi kebijakan kepercayaan peran IAM Anda untuk disertakan `cognito-identity.amazonaws.com`. Konfigurasikan kebijakan kepercayaan peran Anda agar hanya mengizinkan Amazon Cognito mengambil peran saat menampilkan bukti bahwa permintaan tersebut berasal dari pengguna yang diautentikasi di kumpulan identitas spesifik Anda. Untuk informasi selengkapnya, lihat [Kepercayaan peran dan izin](#).
      - C. Pilih Simpan perubahan.
      - D. Untuk mengaktifkan akses tamu, pilih Aktifkan di tab Akses pengguna.

## Mengubah peran yang terkait dengan jenis identitas

Setiap identitas di kolam identitas Anda dikategorikan sebagai diautentikasi atau tidak diautentikasi. Identitas yang diautentikasi adalah milik pengguna yang diautentikasi oleh penyedia login publik (kolam pengguna Amazon Cognito, Login with Amazon, Login dengan Apple, Facebook, Google,



SAML, atau Penyedia OpenID Connect apa pun) atau penyedia developer (proses autentikasi backend Anda sendiri). Identitas yang tidak diautentikasi umumnya adalah milik pengguna tamu.

Untuk setiap jenis identitas, terdapat peran yang ditetapkan. Peran ini memiliki kebijakan yang melekat padanya yang menentukan peran mana Layanan AWS yang dapat diakses. Saat Amazon Cognito menerima permintaan, layanan akan menentukan jenis identitas, menentukan peran yang ditetapkan ke jenis identitas tersebut, dan menggunakan kebijakan yang dilampirkan pada peran tersebut untuk merespons. Dengan memodifikasi kebijakan atau menetapkan peran yang berbeda ke tipe identitas, Anda dapat mengontrol tipe identitas mana Layanan AWS yang dapat diakses. Untuk melihat atau mengubah kebijakan yang terkait dengan peran dalam kolam identitas Anda, lihat [AWS Konsol IAM](#).

Untuk mengubah kumpulan identitas peran default yang diautentikasi atau tidak diautentikasi

1. Pilih kumpulan Identitas dari konsol [Amazon Cognito](#). Pilih kumpulan identitas.
2. Pilih tab Akses pengguna.
3. Temukan akses Tamu atau Akses yang diautentikasi. Dalam kumpulan identitas yang saat ini tidak dikonfigurasi untuk jenis akses tersebut, Status Tidak Aktif. Pilih Edit.
4. Pilih peran IAM default untuk tamu atau pengguna yang diautentikasi di kumpulan identitas Anda.
  - a. Pilih untuk Membuat peran IAM baru jika Anda ingin Amazon Cognito membuat peran baru untuk Anda dengan izin dasar dan hubungan kepercayaan dengan kumpulan identitas Anda. Masukkan nama peran IAM untuk mengidentifikasi peran baru Anda, misalnya `identitypool_authenticatedrole`. Pilih Lihat dokumen kebijakan untuk meninjau izin yang akan ditetapkan Amazon Cognito ke peran IAM baru Anda.
  - b. Anda dapat memilih untuk Menggunakan peran IAM yang ada jika Anda sudah memiliki peran dalam peran Akun AWS yang ingin Anda gunakan. Anda harus mengonfigurasi kebijakan kepercayaan peran IAM Anda untuk disertakan `cognito-identity.amazonaws.com`. Konfigurasikan kebijakan kepercayaan peran Anda agar hanya mengizinkan Amazon Cognito mengambil peran saat menampilkan bukti bahwa permintaan tersebut berasal dari pengguna yang diautentikasi di kumpulan identitas spesifik Anda. Untuk informasi selengkapnya, lihat [Kepercayaan peran dan izin](#).
5. Pilih Simpan perubahan.

## Edit penyedia identitas

Jika Anda mengizinkan pengguna untuk melakukan autentikasi menggunakan penyedia identitas konsumen (misalnya, kumpulan pengguna Amazon Cognito, Login with Amazon, Masuk dengan Apple, Facebook, atau Google), Anda dapat menentukan pengenalan aplikasi di konsol kumpulan identitas Amazon Cognito (identitas federasi). Ini mengaitkan ID aplikasi (disediakan oleh penyedia login publik) dengan kolam identitas Anda.

Anda juga dapat mengonfigurasi aturan autentikasi untuk setiap penyedia dari halaman ini. Setiap penyedia memberikan izin hingga 25 aturan. Aturan diterapkan dalam urutan yang Anda simpan untuk setiap penyedia. Untuk informasi selengkapnya, lihat [Menggunakan kontrol akses berbasis peran](#).

### Warning

Mengubah ID aplikasi iDP yang ditautkan di kumpulan identitas Anda mencegah pengguna yang ada melakukan autentikasi dengan kumpulan identitas tersebut. Untuk informasi selengkapnya, lihat [Identitas mengumpulkan penyedia identitas eksternal](#).

Untuk memperbarui penyedia identitas pool identitas (iDP)

1. Pilih kumpulan Identitas dari konsol [Amazon Cognito](#). Pilih kumpulan identitas.
2. Pilih tab Akses pengguna.
3. Temukan penyedia Identitas. Pilih penyedia identitas yang ingin Anda edit. Jika Anda ingin menambahkan IDP baru, pilih Tambahkan penyedia identitas.
  - Jika Anda memilih Tambahkan penyedia identitas, pilih salah satu jenis Identitas yang ingin Anda tambahkan.
4. Untuk mengubah ID aplikasi, pilih Edit dalam informasi penyedia Identitas.
5. Untuk mengubah peran yang diminta Amazon Cognito saat mengeluarkan kredensial kepada pengguna yang telah diautentikasi dengan penyedia ini, pilih Edit dalam pengaturan Peran.
  - Anda dapat menetapkan pengguna dari IDP tersebut peran Default yang Anda atur saat mengonfigurasi peran Terautentikasi, atau Anda dapat Memilih peran dengan aturan. Dengan IdP kumpulan pengguna Amazon Cognito, Anda juga dapat Memilih peran dengan preferred\_role dalam token. Untuk informasi lebih lanjut tentang cognito:preferred\_role klaim, lihat [Menetapkan nilai prioritas ke grup](#).

- i. Jika Anda memilih Pilih peran dengan aturan, masukkan Klaim sumber dari autentikasi pengguna Anda, Operator yang ingin Anda bandingkan dengan klaim, Nilai yang akan menyebabkan kecocokan dengan pilihan peran ini, dan Peran yang ingin Anda tetapkan saat penetapan Peran cocok. Pilih Tambahkan yang lain untuk membuat aturan tambahan berdasarkan kondisi yang berbeda.
  - ii. Pilih Resolusi Peran. Jika klaim pengguna tidak sesuai dengan aturan, Anda dapat menolak kredensi atau mengeluarkan kredensi untuk peran yang Diautentikasi.
6. Untuk mengubah tag utama yang ditetapkan Amazon Cognito saat mengeluarkan kredensial kepada pengguna yang telah diautentikasi dengan penyedia ini, pilih Edit di Atribut untuk kontrol akses.
  - a. Untuk tidak menerapkan tag utama, pilih Tidak aktif.
  - b. Untuk menerapkan tag utama berdasarkan sub dan aud klaim, pilih Gunakan pemetaan default.
  - c. Untuk membuat skema atribut kustom Anda sendiri ke tag utama, pilih Gunakan pemetaan khusus. Kemudian masukkan kunci Tag yang ingin Anda sumber dari setiap Klaim yang ingin Anda wakili dalam tag.
7. Pilih Simpan perubahan.

## Menghapus kumpulan identitas

Anda tidak dapat membatalkan penghapusan kumpulan identitas. Setelah Anda menghapus kumpulan identitas, semua aplikasi dan pengguna yang bergantung padanya berhenti berfungsi.

Untuk menghapus kolam identitas

1. Pilih kumpulan Identitas dari konsol [Amazon Cognito](#). Pilih tombol radio di sebelah kumpulan identitas yang ingin Anda hapus.
2. Pilih Hapus.
3. Masukkan atau tempel nama kumpulan identitas Anda dan pilih Hapus.

### Warning

Ketika Anda memilih tombol Hapus, Anda akan menghapus kumpulan identitas Anda secara permanen dan semua data pengguna yang dikandungnya. Menghapus kumpulan identitas

akan menyebabkan aplikasi dan layanan lain yang menggunakan kumpulan identitas berhenti bekerja.

## Menghapus identitas dari kumpulan identitas

Saat menghapus identitas dari kumpulan identitas, Anda menghapus informasi identifikasi yang disimpan Amazon Cognito untuk pengguna federasi tersebut. Ketika pengguna Anda meminta kredensi lagi, mereka menerima ID identitas baru jika kumpulan identitas Anda masih mempercayai penyedia identitas mereka. Anda tidak dapat membatalkan operasi ini.

Untuk menghapus identitas

1. Pilih kumpulan Identitas dari konsol [Amazon Cognito](#). Pilih kumpulan identitas.
2. Pilih tab Identity Browser.
3. Pilih kotak centang di samping identitas yang ingin Anda hapus dan pilih Hapus. Konfirmasikan bahwa Anda ingin menghapus identitas dan pilih Hapus.

## Menggunakan Amazon Cognito Sync dengan kumpulan identitas

Amazon Cognito Sync adalah pustaka Layanan AWS dan klien yang memungkinkan untuk menyinkronkan data pengguna terkait aplikasi di seluruh perangkat. Amazon Cognito Sync dapat menyinkronkan data profil pengguna di seluruh perangkat seluler dan web tanpa menggunakan backend Anda sendiri. Pustaka klien menyimpan data secara lokal sehingga aplikasi Anda dapat membaca dan menulis data terlepas dari status konektivitas perangkat. Saat perangkat online, Anda dapat menyinkronkan data. Jika Anda mengatur sinkronisasi push, Anda dapat segera memberi tahu perangkat lain bahwa pembaruan tersedia.

### Mengelola kumpulan data

Jika Anda telah menerapkan fungsionalitas Amazon Cognito Sync di aplikasi Anda, konsol kumpulan identitas Amazon Cognito memungkinkan Anda membuat dan menghapus kumpulan data dan catatan untuk identitas individual secara manual. Perubahan apa pun yang Anda buat pada kumpulan data atau catatan identitas di konsol kumpulan identitas Amazon Cognito tidak akan disimpan hingga Anda memilih Sinkronisasi di konsol. Perubahan tidak terlihat oleh pengguna akhir hingga identitas memanggil Sinkronisasi. Data yang disinkronkan dari perangkat lain untuk identitas individual akan terlihat setelah Anda menyegarkan halaman kumpulan data daftar untuk identitas tertentu.

## Membuat dataset untuk identitas

Amazon Cognito Sync mengaitkan kumpulan data dengan satu identitas. Anda dapat mengisi kumpulan data Anda dengan mengidentifikasi informasi tentang pengguna yang diwakili oleh identitas, lalu menyinkronkan informasi tersebut ke semua perangkat pengguna Anda.

Untuk menambahkan dataset dan catatan dataset ke identitas

1. Pilih kumpulan Identitas dari konsol [Amazon Cognito](#). Pilih kumpulan identitas.
2. Pilih tab Identity Browser.
3. Pilih identitas yang ingin Anda edit.
4. Di Datasets, pilih Create dataset.
5. Masukkan nama Dataset dan pilih Buat kumpulan data.
6. Jika Anda ingin menambahkan catatan ke kumpulan data Anda, pilih kumpulan data Anda dari detail identitas. Di Rekaman, pilih Buat catatan.
7. Masukkan Kunci dan Nilai untuk catatan Anda. Pilih Konfirmasi. Ulangi untuk menambahkan lebih banyak catatan.

Menghapus kumpulan data yang terkait dengan identitas

Untuk menghapus dataset dan catatannya dari identitas

1. Pilih kumpulan Identitas dari konsol [Amazon Cognito](#). Pilih kumpulan identitas.
2. Pilih tab Identity Browser.
3. Pilih identitas yang berisi kumpulan data yang ingin Anda hapus.
4. Di Datasets, pilih tombol radio di sebelah dataset yang ingin Anda hapus.
5. Pilih Hapus. Tinjau pilihan Anda dan pilih Hapus lagi.

## Publikasikan data massal

Publikasi massal dapat digunakan untuk mengekspor data yang sudah disimpan di toko Sinkronisasi Amazon Cognito Anda ke aliran Amazon Kinesis. Untuk memahami petunjuk tentang cara mempublikasikan semua pengaliran Anda secara massal, lihat [Pengaliran Amazon Cognito](#).

## Aktifkan sinkronisasi push

Amazon Cognito secara otomatis melacak hubungan antara identitas dan perangkat. Dengan menggunakan fitur push sync, Anda dapat memastikan bahwa setiap instance dari identitas yang diberikan diberi tahu saat data identitas berubah. Sinkronisasi push membuatnya sehingga, setiap kali dataset berubah untuk identitas, semua perangkat yang terkait dengan identitas itu menerima pemberitahuan push senyap yang memberi tahu mereka tentang perubahan tersebut.

Anda dapat mengaktifkan sinkronisasi push di konsol Amazon Cognito.

Untuk mengaktifkan sinkronisasi push

1. Pilih kumpulan Identitas dari konsol [Amazon Cognito](#). Pilih kumpulan identitas.
2. Pilih tab Properti kolam Identity.
3. Dalam sinkronisasi push, pilih Edit
4. Pilih Aktifkan sinkronisasi push dengan kumpulan identitas Anda.
5. Pilih salah satu aplikasi Platform Amazon Simple Notification Service (Amazon SNS) Platform yang Anda buat saat ini. Wilayah AWS Amazon Cognito menerbitkan pemberitahuan push ke aplikasi platform Anda. Pilih Buat aplikasi platform untuk menavigasi ke konsol Amazon SNS dan membuat yang baru.
6. Untuk mempublikasikan ke aplikasi platform Anda, Amazon Cognito mengasumsikan peran IAM dalam aplikasi Anda. Akun AWS Pilih untuk Membuat peran IAM baru jika Anda ingin Amazon Cognito membuat peran baru untuk Anda dengan izin dasar dan hubungan kepercayaan dengan kumpulan identitas Anda. Masukkan nama peran IAM untuk mengidentifikasi peran baru Anda, misalnya `myidentitypool1_authenticatedrole`. Pilih Lihat dokumen kebijakan untuk meninjau izin yang akan ditetapkan Amazon Cognito ke peran IAM baru Anda.
7. Anda dapat memilih untuk Menggunakan peran IAM yang ada jika Anda sudah memiliki peran dalam peran Akun AWS yang ingin Anda gunakan. Anda harus mengonfigurasi kebijakan kepercayaan peran IAM Anda untuk disertakan `cognito-identity.amazonaws.com`. Konfigurasi kebijakan kepercayaan peran Anda agar hanya mengizinkan Amazon Cognito mengambil peran saat menampilkan bukti bahwa permintaan tersebut berasal dari pengguna yang diautentikasi di kumpulan identitas spesifik Anda. Untuk informasi selengkapnya, lihat [Kepercayaan peran dan izin](#).
8. Pilih Simpan perubahan.

## Siapkan Amazon Cognito Streams

Pengaliran Amazon Cognito memberikan developer kontrol dan wawasan terkait data mereka yang disimpan di Amazon Cognito Sync. Developer sekarang dapat mengkonfigurasi pengaliran Kinesis untuk menerima peristiwa sebagai data. Amazon Cognito dapat mendorong setiap perubahan set data ke pengaliran Kinesis yang Anda miliki secara langsung. Untuk mengetahui petunjuk tentang cara mengatur Pengaliran Amazon Cognito di konsol Amazon Cognito, lihat [Pengaliran Amazon Cognito](#).

## Siapkan Acara Amazon Cognito

Acara Amazon Cognito memungkinkan Anda menjalankan AWS Lambda fungsi sebagai respons terhadap peristiwa penting di Amazon Cognito Sync. Amazon Cognito Sync memunculkan peristiwa Sync Trigger ketika set data disinkronkan. Anda dapat menggunakan peristiwa Sync Trigger untuk mengambil tindakan ketika pengguna memperbarui data. Untuk mengetahui instruksi tentang menyiapkan Peristiwa Amazon Cognito dari konsol, lihat [Peristiwa Amazon Cognito](#).

Untuk mempelajari lebih lanjut tentang AWS Lambda, lihat [AWS Lambda](#).

## Konsep kolam identitas

Anda dapat menggunakan kumpulan identitas Amazon Cognito untuk membuat identitas unik bagi pengguna Anda dan mengautentikasi mereka dengan penyedia identitas. Dengan identitas, Anda dapat memperoleh AWS kredensi hak istimewa terbatas sementara untuk mengakses lainnya. Layanan AWS Kumpulan identitas Amazon Cognito mendukung penyedia identitas publik—Amazon, Apple, Facebook, dan Google—serta identitas yang tidak diautentikasi. Ia juga mendukung identitas yang di-autentikasi developer, yang memungkinkan Anda mendaftar dan melakukan autentikasi pengguna melalui proses autentikasi backend Anda sendiri.

Untuk informasi tentang ketersediaan Wilayah kolam identitas Amazon Cognito, lihat [AWS Ketersediaan Wilayah Layanan](#). Untuk informasi selengkapnya tentang konsep kolam identitas Amazon Cognito, lihat topik-topik berikut.

### Topik

- [Aliran otentikasi kumpulan identitas \(identitas federasi\)](#)
- [Peran IAM](#)
- [Kepercayaan peran dan izin](#)

## Aliran otentikasi kumpulan identitas (identitas federasi)

Amazon Cognito membantu Anda membuat pengidentifikasi unik untuk pengguna akhir Anda yang tetap konsisten di seluruh perangkat dan platform. Amazon Cognito juga memberikan kredensial hak istimewa terbatas sementara ke aplikasi Anda untuk mengakses sumber daya. AWS Halaman ini membahas dasar-dasar cara kerja otentikasi di Amazon Cognito dan menjelaskan siklus hidup identitas di dalam kumpulan identitas Anda.

### Authflow penyedia eksternal

Seorang pengguna yang mengautentikasi dengan Amazon Cognito melalui proses multi-langkah untuk mem-bootstrap kredensialnya. Amazon Cognito memiliki dua alur berbeda untuk autentikasi dengan penyedia publik: ditingkatkan dan dasar.

Setelah menyelesaikan salah satu alur ini, Anda dapat mengakses aliran lain Layanan AWS seperti yang ditentukan oleh kebijakan akses peran Anda. Secara default, [konsol Amazon Cognito](#) membuat peran dengan akses ke toko Amazon Cognito Sync dan ke Amazon Mobile Analytics. Untuk informasi selengkapnya tentang cara memberikan akses tambahan, lihat [Peran IAM](#).

Identity pool menerima artefak berikut dari penyedia:

Penyedia	Artefak otentikasi
Kolam pengguna Amazon Cognito	Token ID
OpenID Connect (OIDC)	Token ID
SAML 2.0	Pernyataan SAFL
Penyedia sosial	Token akses

### Authflow yang disempurnakan (disederhanakan)

Saat Anda menggunakan authflow yang disempurnakan, aplikasi Anda terlebih dahulu menampilkan bukti autentikasi dari kumpulan pengguna Amazon Cognito resmi atau penyedia identitas pihak ketiga dalam permintaan. [GetId](#)

1. [Aplikasi Anda menyajikan bukti otentikasi—token web JSON atau pernyataan SAML—dari kumpulan pengguna Amazon Cognito resmi atau penyedia identitas pihak ketiga dalam permintaan getId.](#)

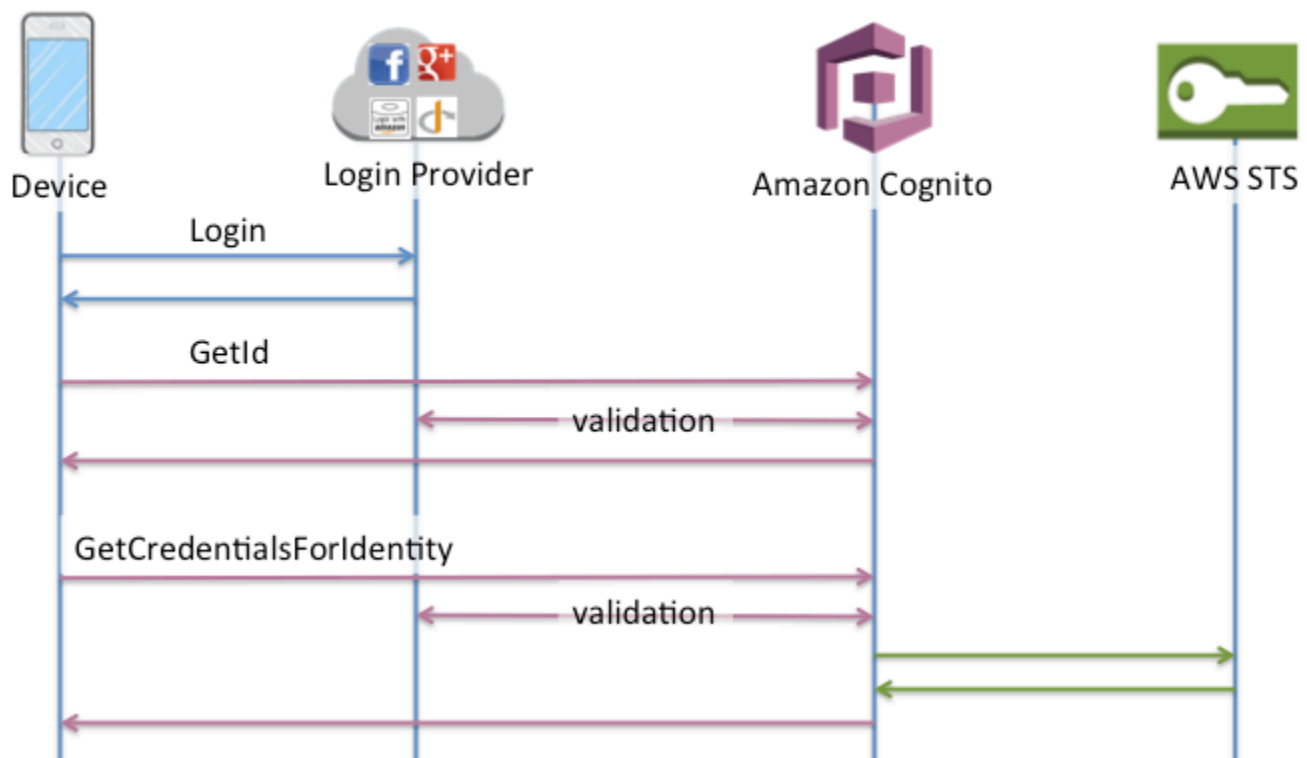


2. Kumpulan identitas Anda mengembalikan ID identitas.
3. Aplikasi Anda menggabungkan ID identitas dengan bukti otentikasi yang sama dalam [GetCredentialsForIdentity](#) permintaan.
4. Kumpulan identitas Anda mengembalikan AWS kredensialnya.
5. Aplikasi Anda menandatangani permintaan AWS API dengan kredensi sementara.

Otentikasi yang disempurnakan mengelola logika pemilihan peran IAM dan pengambilan kredensial dalam konfigurasi kumpulan identitas Anda. Anda dapat mengonfigurasi kumpulan identitas untuk memilih peran default, untuk menerapkan prinsip kontrol akses berbasis atribut (ABAC) atau kontrol akses berbasis peran (RBAC) ke pemilihan peran. AWS Kredensi dari otentikasi yang disempurnakan berlaku selama satu jam.

Urutan operasi dalam otentikasi yang disempurnakan

1. GetId
2. GetCredentialsForIdentity



Authflow dasar (klasik)

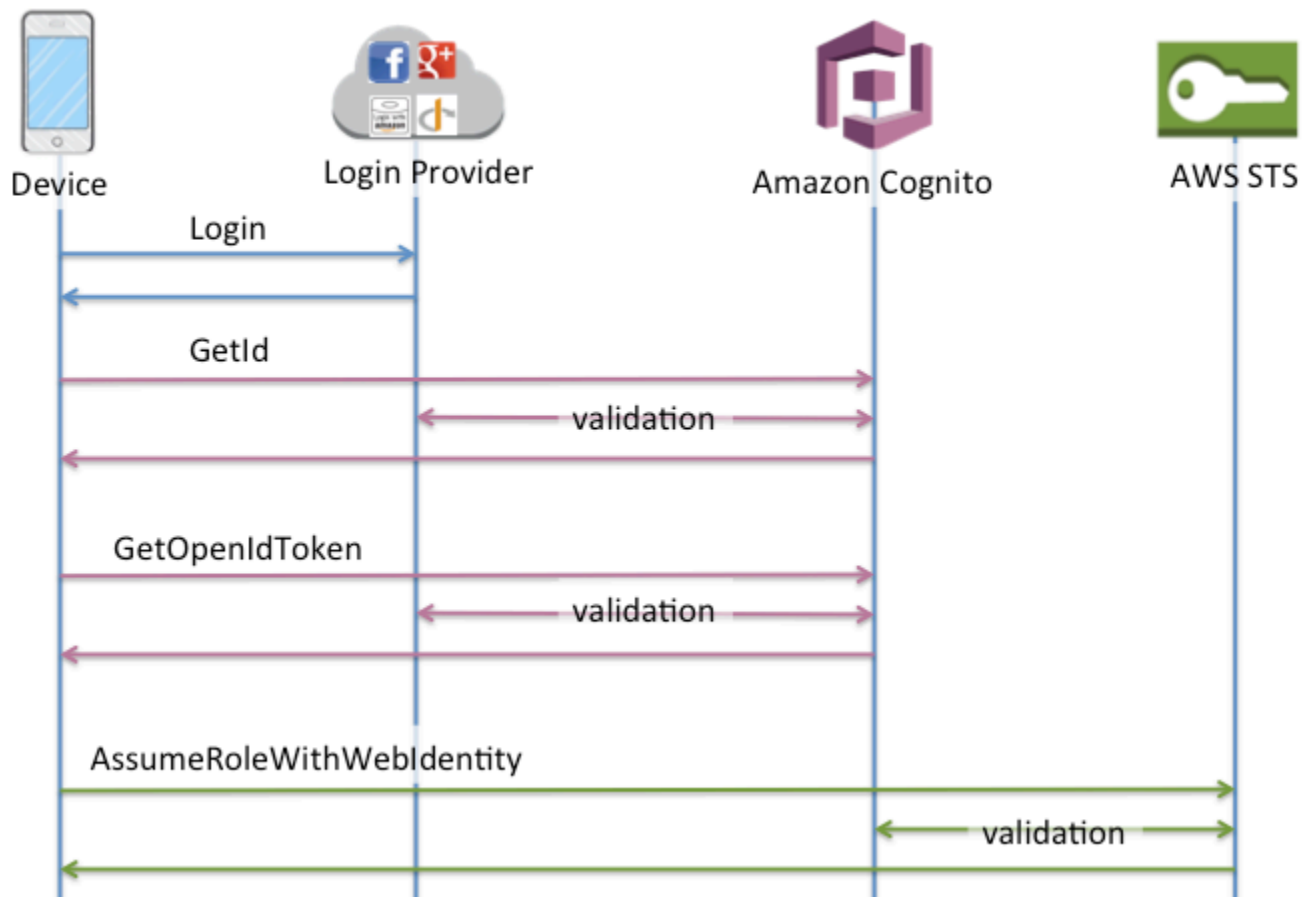
Saat Anda menggunakan authflow dasar,

1. [Aplikasi Anda menyajikan bukti otentikasi—token web JSON atau pernyataan SAML—dari kumpulan pengguna Amazon Cognito resmi atau penyedia identitas pihak ketiga dalam permintaan getId.](#)
2. Kumpulan identitas Anda mengembalikan ID identitas.
3. Aplikasi Anda menggabungkan ID identitas dengan bukti otentikasi yang sama dalam [GetOpenIdToken](#) permintaan.
4. [GetOpenIdToken](#) mengembalikan token OAuth 2.0 baru yang dikeluarkan oleh kumpulan identitas Anda.
5. Aplikasi Anda menyajikan token baru dalam [AssumeRoleWithWebIdentity](#) permintaan.
6. AWS Security Token Service (AWS STS) mengembalikan AWS kredensi.
7. Aplikasi Anda menandatangani permintaan AWS API dengan kredensi sementara.

Alur kerja dasar memberi Anda kontrol yang lebih terperinci atas kredensial yang Anda distribusikan ke pengguna Anda. [GetCredentialsForIdentity](#) permintaan authflow yang disempurnakan meminta peran berdasarkan konten token akses. [AssumeRoleWithWebIdentity](#) permintaan dalam alur kerja klasik memberi aplikasi Anda kemampuan yang lebih besar untuk meminta kredensial untuk AWS Identity and Access Management peran apa pun yang telah Anda konfigurasi dengan kebijakan kepercayaan yang memadai. Anda juga dapat meminta durasi sesi peran khusus.

Urutan operasi dalam otentikasi Dasar

1. `GetId`
2. `GetOpenIdToken`
3. `AssumeRoleWithWebIdentity`



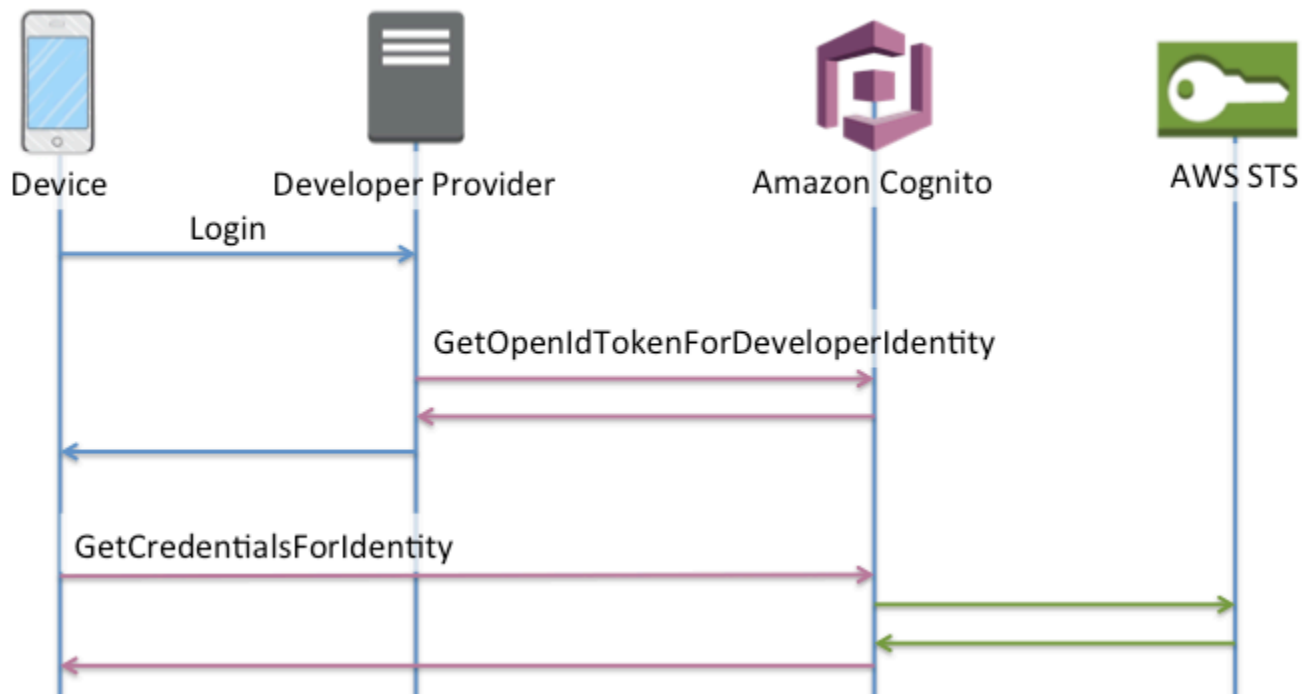
## Identitas otentikasi pengembang authflow

Saat menggunakan [Identitas yang diautentikasi pengembang \(kumpulan identitas\)](#), klien menggunakan authflow berbeda yang menyertakan kode di luar Amazon Cognito untuk memvalidasi pengguna di sistem autentikasi Anda sendiri. Kode di luar Amazon Cognito ditunjukkan seperti itu.

## Authflow yang disempurnakan

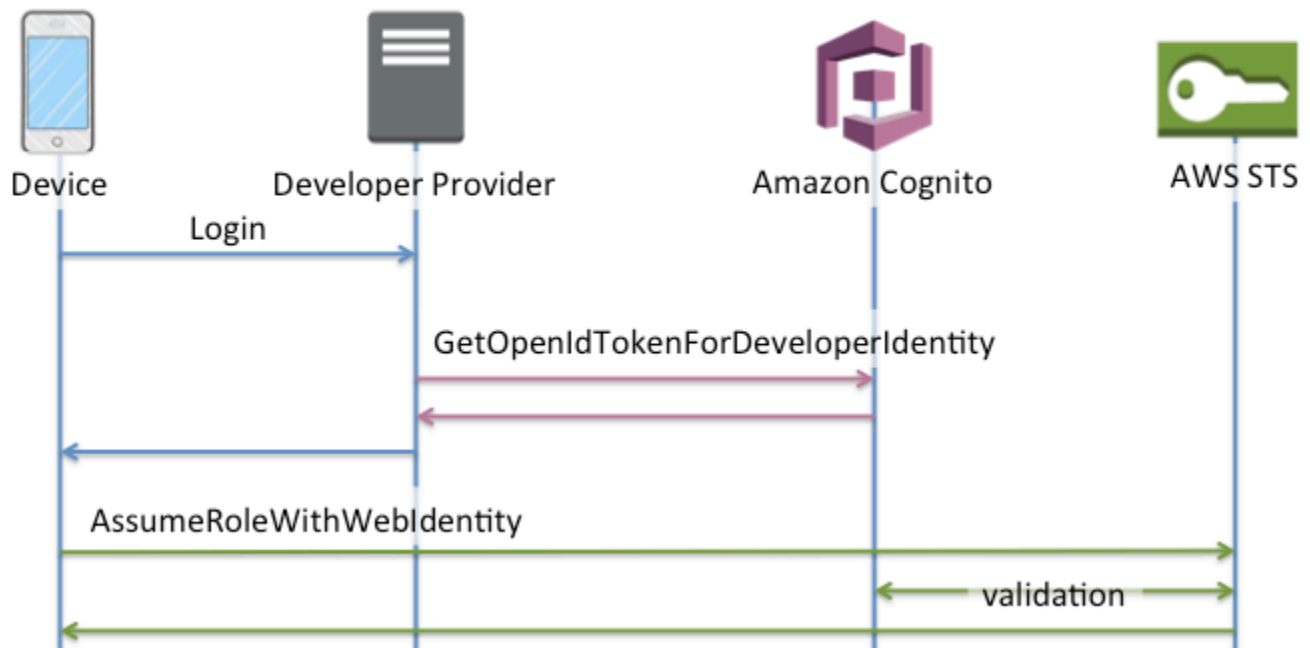
Urutan operasi dalam otentikasi yang ditingkatkan dengan penyedia pengembang

1. Login melalui Penyedia Developer (kode di luar Amazon Cognito)
2. Validasi login pengguna (kode di luar Amazon Cognito)
3. [GetOpenIdTokenForDeveloperIdentity](#)
4. [GetCredentialsForIdentity](#)



Urutan operasi dalam otentikasi Dasar dengan penyedia pengembang

1. Terapkan logika di luar kumpulan identitas untuk masuk dan menghasilkan pengenal pengembang-penyedia.
2. Ambil kredensial-sisi server AWS yang tersimpan.
3. Kirim pengenal penyedia pengembang dalam permintaan [GetOpenIdTokenForDeveloperIdentity](#) API yang ditandatangani dengan AWS kredensial resmi.
4. Minta kredensi aplikasi dengan. [AssumeRoleWithWebIdentity](#)



Authflow mana yang harus saya gunakan?

Aliran yang disempurnakan adalah pilihan paling aman dengan tingkat upaya pengembang terendah:

- Alur yang disempurnakan mengurangi kompleksitas, ukuran, dan laju permintaan API.
- Aplikasi Anda tidak perlu membuat permintaan API tambahan AWS STS.
- Kumpulan identitas Anda mengevaluasi pengguna Anda untuk kredensi peran IAM yang harus mereka terima. Anda tidak perlu menanamkan logika untuk pemilihan peran di klien Anda.

#### ⚠ Important

Saat Anda membuat kumpulan identitas baru, jangan aktifkan otentikasi dasar (klasik) secara default, sebagai praktik terbaik. Untuk menerapkan otentikasi dasar, pertama-tama evaluasi hubungan kepercayaan peran IAM Anda untuk identitas web. Kemudian bangun logika untuk pemilihan peran ke klien Anda dan amankan klien dari modifikasi oleh pengguna.

Alur otentikasi dasar mendelegasikan logika pemilihan peran IAM ke aplikasi Anda. Dalam alur ini, Amazon Cognito memvalidasi sesi yang diautentikasi atau tidak diautentikasi pengguna Anda dan mengeluarkan token yang dapat Anda tukarkan dengan kredensialnya. AWS STS Pengguna dapat menukar token dari otentikasi dasar untuk setiap peran IAM yang mempercayai kumpulan identitas Anda dan `amr`, atau status yang diautentikasi/tidak diautentikasi.

Demikian pula, pahami bahwa otentikasi pengembang adalah jalan pintas seputar validasi otentikasi penyedia identitas. Amazon Cognito mempercayai AWS kredensial yang mengotorisasi [GetOpenIdTokenForDeveloperIdentity](#) permintaan tanpa validasi tambahan dari konten permintaan. Amankan rahasia yang mengotorisasi otentikasi pengembang dari akses oleh pengguna.

## Ringkasan API

### GetId

Panggilan [GetId](#) API adalah panggilan pertama yang diperlukan untuk membangun identitas baru di Amazon Cognito.

#### Akses tidak diautentikasi

Amazon Cognito dapat memberikan akses tamu yang tidak diautentikasi di aplikasi Anda. Jika fitur ini diaktifkan di kumpulan identitas Anda, pengguna dapat meminta ID identitas baru kapan saja melalui [GetId](#) API. Aplikasi diharapkan meng-cache ID identitas ini untuk melakukan panggilan berikutnya ke Amazon Cognito. SDK AWS Seluler dan AWS SDK untuk JavaScript di Browser memiliki penyedia kredensi yang menangani caching ini untuk Anda.

#### Akses yang diautentikasi

Ketika Anda telah mengonfigurasi aplikasi Anda dengan dukungan untuk penyedia login publik (Facebook, Google+, Login with Amazon, atau Masuk dengan Apple), pengguna juga dapat menyediakan token (OAuth atau OpenID Connect) yang mengidentifikasi mereka di penyedia tersebut. Saat digunakan dalam panggilan ke [GetId](#), Amazon Cognito membuat identitas baru yang diautentikasi atau mengembalikan identitas yang sudah dikaitkan dengan login tertentu. Amazon Cognito melakukan ini dengan memvalidasi token dengan penyedia dan memastikan hal-hal berikut:

- Token valid dan dari penyedia yang dikonfigurasi.
- Token tidak kedaluwarsa.
- Token cocok dengan pengenalan aplikasi yang dibuat dengan penyedia itu (misalnya, ID aplikasi Facebook).
- Token cocok dengan pengenalan pengguna.

### GetCredentialsForIdentity

[GetCredentialsForIdentity](#) API dapat dipanggil setelah Anda membuat ID identitas. Operasi ini secara fungsional setara dengan memanggil [GetOpenIdToken](#), kemudian [AssumeRoleWithWebIdentity](#).

Agar Amazon Cognito dapat menelepon `AssumeRoleWithWebIdentity` atas nama Anda, kumpulan identitas Anda harus memiliki peran IAM yang terkait dengannya. Anda dapat melakukan ini melalui konsol Amazon Cognito atau secara manual melalui operasi.

[SetIdentityPoolRoles](#)

## GetOpenIdToken

Buat permintaan [GetOpenIdToken](#) API setelah Anda membuat ID identitas. Cache ID identitas setelah permintaan pertama Anda, dan mulai sesi dasar (klasik) berikutnya untuk identitas tersebut `GetOpenIdToken`.

Respons terhadap permintaan `GetOpenIdToken` API adalah token yang dihasilkan Amazon Cognito. Anda dapat mengirimkan token ini sebagai `WebIdentityToken` parameter dalam [AssumeRoleWithWebIdentity](#) permintaan.

Sebelum Anda mengirimkan token OpenID, verifikasi di aplikasi Anda. Anda dapat menggunakan pustaka OIDC di SDK atau pustaka seperti [aws-jwt-verify](#) untuk mengonfirmasi bahwa Amazon Cognito mengeluarkan token. [ID kunci penandatanganan, atau `id`, dari token OpenID adalah salah satu yang tercantum dalam dokumen Identitas Amazon Cognito `ƒjwks\_uri`](#). Kunci-kunci ini dapat berubah. Fungsi Anda yang memverifikasi token Identitas Amazon Cognito harus memperbarui daftar kuncinya secara berkala dari dokumen `jwks_uri`. Amazon Cognito menetapkan durasi penyegaran di header respons kontrol cache `jwks_uri`, yang saat ini disetel ke 30 hari. `max-age`

### Akses tidak diautentikasi

Untuk mendapatkan token untuk identitas yang tidak diautentikasi, Anda hanya memerlukan ID identitas itu sendiri. Tidak mungkin mendapatkan token yang tidak diautentikasi untuk identitas atau identitas yang diautentikasi yang telah Anda nonaktifkan.

### Akses yang diautentikasi

Jika Anda memiliki identitas yang diautentikasi, Anda harus memberikan setidaknya satu token yang valid untuk login yang telah dikaitkan dengan identitas tersebut. Semua token yang diteruskan selama `GetOpenIdToken` panggilan harus melewati validasi yang sama yang disebutkan sebelumnya; jika salah satu token gagal, seluruh panggilan gagal. Tanggapan dari `GetOpenIdToken` panggilan juga mencakup ID identitas. Ini karena ID identitas yang Anda lewati mungkin bukan yang dikembalikan.

## Menautkan login

Jika Anda mengirimkan token untuk login yang belum dikaitkan dengan identitas apa pun, login dianggap “ditautkan” ke identitas terkait. Anda hanya dapat menautkan satu login per penyedia publik. Upaya untuk menautkan lebih dari satu login dengan penyedia publik menghasilkan respons `ResourceConflictException` kesalahan. Jika login hanya ditautkan ke identitas yang ada, ID identitas `GetOpenIdToken` yang dikembalikan sama dengan yang Anda lewatkan.

## Menggabungkan identitas

Jika Anda memberikan token untuk login yang saat ini tidak ditautkan ke identitas yang diberikan, tetapi ditautkan ke identitas lain, kedua identitas tersebut akan digabungkan. Setelah digabungkan, satu identitas menjadi induk/pemilik semua login yang terkait dan identitas lainnya dinonaktifkan. Dalam hal ini, ID identitas dari induk/pemilik dikembalikan. Anda harus memperbarui cache lokal Anda jika nilai ini berbeda. Penyedia di SDK AWS Seluler atau AWS SDK untuk JavaScript di Browser melakukan operasi ini untuk Anda.

## `GetOpenIdTokenForDeveloperIdentity`

[GetOpenIdTokenForDeveloperIdentity](#) Operasi menggantikan penggunaan [GetId](#) dan [GetOpenIdToken](#) dari perangkat saat menggunakan identitas otentikasi pengembang. Karena aplikasi Anda menandatangani permintaan ke operasi API ini dengan AWS kredensial, Amazon Cognito percaya bahwa pengenalan pengguna yang disediakan dalam permintaan tersebut valid. Otentikasi pengembang menggantikan validasi token yang dilakukan Amazon Cognito dengan penyedia eksternal.

Payload untuk API ini mencakup `logins` peta. Peta ini harus berisi kunci penyedia pengembang Anda dan nilai sebagai pengenalan bagi pengguna di sistem Anda. Jika pengenalan pengguna belum ditautkan ke identitas yang ada, Amazon Cognito akan membuat identitas baru dan mengembalikan ID identitas baru dan token OpenID Connect untuk identitas tersebut. Jika pengenalan pengguna sudah ditautkan, Amazon Cognito mengembalikan ID identitas yang sudah ada sebelumnya dan token OpenID Connect. Cache ID identitas pengembang setelah permintaan pertama Anda, dan mulai sesi dasar (klasik) berikutnya untuk identitas tersebut `GetOpenIdTokenForDeveloperIdentity`.

Respons terhadap permintaan `GetOpenIdTokenForDeveloperIdentity` API adalah token yang dihasilkan Amazon Cognito. Anda dapat mengirimkan token ini sebagai `WebIdentityToken` parameter dalam `AssumeRoleWithWebIdentity` permintaan.



Sebelum Anda mengirimkan token OpenID Connect, verifikasi di aplikasi Anda. Anda dapat menggunakan pustaka OIDC di SDK atau pustaka seperti [aws-jwt-verify](#) untuk mengonfirmasi bahwa Amazon Cognito mengeluarkan token. ID kunci penandatanganan, atau `kid`, dari token OpenID Connect adalah salah satu yang tercantum dalam dokumen `jwtks_uri` Amazon Cognito [Identity](#) †. Kunci-kunci ini dapat berubah. Fungsi Anda yang memverifikasi token Identitas Amazon Cognito harus memperbarui daftar kuncinya secara berkala dari dokumen `jwtks_uri`. Amazon Cognito menetapkan durasi penyegaran di header `cache-control` respons `jwtks_uri`, yang saat ini disetel ke 30 hari. `max-age`

### Menautkan login

Sebagai penyedia eksternal, menyediakan login tambahan yang belum dikaitkan dengan identitas secara implisit menghubungkan login tersebut ke identitas tersebut. Jika Anda menautkan login penyedia eksternal ke identitas, pengguna dapat menggunakan authflow penyedia eksternal dengan penyedia tersebut. Namun, mereka tidak dapat menggunakan nama penyedia pengembang Anda di peta login saat menelepon `GetId` atau `GetOpenIdToken`.

### Menggabungkan identitas

Dengan identitas otentikasi pengembang, Amazon Cognito mendukung penggabungan implisit dan penggabungan eksplisit melalui API. [MergeDeveloperIdentities](#) Dengan penggabungan eksplisit, Anda dapat menandai dua identitas dengan pengidentifikasi pengguna di sistem Anda sebagai identitas tunggal. Jika Anda menyediakan pengenalan pengguna sumber dan tujuan, Amazon Cognito menggabungkannya. Lain kali Anda meminta token OpenID Connect untuk salah satu pengenalan pengguna, id identitas yang sama akan dikembalikan.

### AssumeRoleWithWebIdentity

Setelah Anda memiliki token OpenID Connect, Anda kemudian dapat menukar ini untuk AWS kredensi sementara melalui permintaan [AssumeRoleWithWebIdentity](#) API ke (). AWS Security Token Service AWS STS

Karena tidak ada batasan jumlah identitas yang dapat Anda buat, penting untuk memahami izin yang Anda berikan kepada pengguna Anda. Siapkan peran IAM yang berbeda untuk aplikasi Anda: satu untuk pengguna yang tidak diautentikasi, dan satu untuk pengguna yang diautentikasi. Konsol Amazon Cognito dapat membuat peran default saat pertama kali mengatur kumpulan identitas Anda. Peran ini secara efektif tidak memiliki izin yang diberikan. Ubah mereka untuk memenuhi kebutuhan Anda.

Pelajari lebih lanjut tentang [Kepercayaan peran dan izin](#).

† Dokumen [jwks\\_uri](#) Identitas Amazon Cognito default berisi informasi tentang kunci yang menandatangani token untuk kumpulan identitas di sebagian besar. Wilayah AWS Daerah berikut memiliki dokumen `jwks_uri` yang berbeda.

### Amazon Cognito Identity JSON web key URIs in other Wilayah AWS

Wilayah AWS	Jalur ke dokumen <code>jwks_uri</code>
AWS GovCloud (AS-Barat)	<code>https://cognito-identity.us-gov-west-1.amazonaws.com/.well-known/jwks_uri</code>
Tiongkok (Beijing)	<code>https://cognito-identity.cn-north-1.amazonaws.com.cn/.well-known/jwks_uri</code>
Opt-in Regions seperti Eropa (Milan) dan Afrika (Cape Town)	<code>https://cognito-identity. <i>Region</i>.amazonaws.com/.well-known/jwks_uri</code>

Anda juga dapat mengekstrapolasi `jwks_uri` dari penerbit atau yang `iss` Anda terima di token OpenID dari Amazon Cognito. Titik akhir penemuan standar OIDC `<issuer>/.well-known/openid-configuration` mencantumkan jalur ke `jwks_uri` untuk token Anda.

## Peran IAM

Saat membuat kumpulan identitas, Anda diminta untuk memperbarui peran IAM yang diasumsikan pengguna Anda. Peran IAM berfungsi seperti ini: Saat pengguna masuk ke aplikasi Anda, Amazon Cognito akan menghasilkan kredensi AWS sementara untuk pengguna. Kredensial sementara ini terkait dengan IAM role tertentu. Dengan peran IAM, Anda dapat menentukan serangkaian izin untuk mengakses sumber daya Anda AWS .

Anda dapat menentukan IAM role default untuk pengguna yang terautentikasi dan tidak terautentikasi. Selain itu, Anda dapat menentukan aturan untuk memilih peran untuk setiap pengguna berdasarkan klaim di token ID pengguna. Untuk informasi selengkapnya, lihat [Menggunakan kontrol akses berbasis peran](#).

Secara default, konsol Amazon Cognito membuat peran IAM yang menyediakan akses ke Amazon Mobile Analytics dan ke Amazon Cognito Sync. Atau, Anda dapat memilih untuk menggunakan IAM role yang sudah ada.

Ubah peran IAM untuk mengizinkan atau membatasi akses ke layanan lain. Untuk melakukannya, [masuk ke Konsol IAM](#). Kemudian pilih Peran, dan pilih peran. Kebijakan yang dilampirkan pada peran yang dipilih tercantum di tab Izin. Anda dapat menyesuaikan kebijakan akses dengan memilih tautan Kelola Kebijakan yang sesuai. Untuk mempelajari lebih lanjut tentang menggunakan dan menentukan kebijakan, lihat [Ikhtisar Kebijakan IAM](#).

#### Note

Sebagai praktik terbaik, tentukan kebijakan yang mengikuti prinsip pemberian hak istimewa minimal. Dengan kata lain, kebijakan mencakup hanya izin yang diperlukan pengguna untuk melakukan tugas-tugas mereka. Untuk informasi lebih lanjut, lihat [Memberikan Hak Akses Minimal](#) dalam Panduan Pengguna IAM.

Ingat bahwa identitas yang tidak diautentikasi diasumsikan oleh pengguna yang tidak masuk ke aplikasi Anda. Biasanya, izin yang Anda tetapkan untuk identitas yang tidak diautentikasi harus lebih ketat daripada untuk identitas diautentikasi.

#### Topik

- [Menyiapkan kebijakan kepercayaan](#)
- [Kebijakan akses](#)

### Menyiapkan kebijakan kepercayaan

Amazon Cognito menggunakan peran IAM untuk menghasilkan kredensial sementara bagi pengguna aplikasi Anda. Akses ke izin dikendalikan oleh hubungan kepercayaan peran. Pelajari selengkapnya tentang [Kepercayaan peran dan izin](#).

Token yang disajikan AWS STS dihasilkan oleh kumpulan identitas, yang menerjemahkan kumpulan pengguna, sosial, atau token penyedia OIDC, atau pernyataan SAFL, ke tokennya sendiri. Token kumpulan identitas berisi aud klaim yang merupakan ID kumpulan identitas.

Contoh kebijakan kepercayaan peran berikut memungkinkan prinsipal layanan federasi `cognito-identity.amazonaws.com` untuk memanggil `AWS STS API AssumeRoleWithWebIdentity`.

Permintaan hanya akan berhasil jika token kumpulan identitas dalam permintaan API memiliki klaim berikut.

1. `aud` klaim ID kumpulan identitas `us-west-2:abcdefg-1234-5678-910a-0e8443553f95`.
2. `amr` klaim `authenticated` tersebut ditambahkan saat pengguna masuk dan bukan pengguna tamu.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "cognito-identity.amazonaws.com"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "cognito-identity.amazonaws.com:aud": "us-west-2:abcdefg-1234-5678-910a-0e8443553f95"
        },
        "ForAnyValue:StringLike": {
          "cognito-identity.amazonaws.com:amr": "authenticated"
        }
      }
    }
  ]
}
```

### Kebijakan kepercayaan untuk peran IAM dalam otentikasi Dasar (Klasik)

Anda harus menerapkan setidaknya satu kondisi yang membatasi kebijakan kepercayaan untuk peran yang Anda gunakan dengan kumpulan identitas. Saat Anda membuat atau memperbarui kebijakan kepercayaan peran untuk kumpulan identitas, IAM akan menampilkan kesalahan jika Anda mencoba menyimpan perubahan tanpa setidaknya satu kunci kondisi yang membatasi identitas sumber. AWS STS tidak mengizinkan [AssumeRoleWithWebIdentity](#) operasi lintas akun dari kumpulan identitas ke peran IAM yang tidak memiliki kondisi jenis ini.

Topik ini mencakup beberapa kondisi yang membatasi identitas sumber untuk kumpulan identitas. Untuk daftar lengkap, lihat [Kunci yang tersedia untuk federasi identitas AWS web](#).

Dalam otentikasi dasar, atau klasik, dengan kumpulan identitas, Anda dapat mengambil peran IAM apa pun AWS STS jika memiliki kebijakan kepercayaan yang tepat. Peran IAM untuk kumpulan identitas Amazon Cognito mempercayai `cognito-identity.amazonaws.com` kepala layanan untuk mengambil peran tersebut. Konfigurasi ini tidak cukup untuk mengamankan peran IAM Anda terhadap akses yang tidak diinginkan ke sumber daya. Peran jenis ini harus menerapkan kondisi tambahan pada kebijakan kepercayaan peran. Anda tidak dapat membuat atau memodifikasi peran untuk kumpulan identitas tanpa setidaknya satu dari kondisi berikut.

#### **`cognito-identity.amazonaws.com:aud`**

Membatasi peran untuk operasi dari satu atau lebih kumpulan identitas. Amazon Cognito menunjukkan kumpulan identitas sumber dalam `aud` klaim dalam token kumpulan identitas.

#### **`cognito-identity.amazonaws.com:amr`**

Membatasi peran untuk salah satu `authenticated` atau `unauthenticated` (tamu) pengguna. Amazon Cognito menunjukkan status otentikasi dalam `amr` klaim di token kumpulan identitas.

#### **`cognito-identity.amazonaws.com:sub`**

Membatasi peran untuk satu atau lebih pengguna oleh UUID. UUID ini adalah ID identitas pengguna di kolam identitas. Nilai ini bukan sub nilai dari penyedia identitas asli pengguna. Amazon Cognito menunjukkan UUID ini dalam `sub` klaim di token kumpulan identitas.

Autentikasi aliran yang ditingkatkan mengharuskan peran IAM Akun AWS sama dengan kumpulan identitas, tetapi ini tidak terjadi dalam otentikasi dasar.

Pertimbangan tambahan berlaku untuk kumpulan identitas Amazon Cognito yang mengambil peran IAM lintas [akun](#). Kebijakan kepercayaan dari peran tersebut harus menerima prinsip `cognito-identity.amazonaws.com` layanan dan harus mengandung `cognito-identity.amazonaws.com:aud` kondisi spesifik. Untuk mencegah akses yang tidak diinginkan ke AWS sumber daya Anda, kunci `aud` kondisi membatasi peran pengguna dari kumpulan identitas dalam nilai kondisi.

Token yang dikeluarkan oleh kumpulan identitas untuk identitas berisi informasi tentang Akun AWS asal kumpulan identitas. Saat Anda menampilkan token kumpulan identitas dalam permintaan [AssumeRoleWithWebIdentity](#) API, AWS STS memeriksa apakah kumpulan identitas asal Akun AWS sama dengan peran IAM. Jika AWS STS menentukan bahwa permintaan tersebut lintas akun, ia memeriksa untuk melihat apakah kebijakan kepercayaan peran memiliki `aud` kondisi. Panggilan peran asumsi gagal jika tidak ada kondisi seperti itu dalam kebijakan kepercayaan peran. Jika

permintaan tidak lintas akun, AWS STS tidak memberlakukan pembatasan ini. Sebagai praktik terbaik, selalu terapkan kondisi jenis ini pada kebijakan kepercayaan peran kumpulan identitas Anda.

### Ketentuan kebijakan kepercayaan tambahan

Gunakan kembali peran di seluruh kumpulan identitas

Untuk menggunakan kembali peran di seluruh kolam identitas, karena mereka berbagi serangkaian izin umum, Anda dapat menyertakan beberapa kolam identitas, seperti ini:

```
"StringEquals": {
  "cognito-identity.amazonaws.com:aud": [
    "us-east-1:12345678-abcd-abcd-abcd-123456790ab",
    "us-east-1:98765432-dcba-dcba-dcba-123456790ab"
  ]
}
```

### Batasi akses ke identitas tertentu

Untuk membuat kebijakan terbatas pada serangkaian pengguna aplikasi tertentu, periksa nilai `cognito-identity.amazonaws.com:sub`:

```
"StringEquals": {
  "cognito-identity.amazonaws.com:aud": "us-east-1:12345678-abcd-abcd-abcd-123456790ab",
  "cognito-identity.amazonaws.com:sub": [
    "us-east-1:12345678-1234-1234-1234-123456790ab",
    "us-east-1:98765432-1234-1234-1243-123456790ab"
  ]
}
```

### Batasi akses ke penyedia tertentu

Untuk membuat kebijakan terbatas untuk pengguna yang telah login dengan penyedia tertentu (mungkin penyedia login Anda sendiri), periksa nilai `cognito-identity.amazonaws.com:amr`:

```
"ForAnyValue:StringLike": {
  "cognito-identity.amazonaws.com:amr": "login.myprovider.myapp"
}
```

Sebagai contoh, sebuah aplikasi yang hanya mempercayai Facebook akan memiliki klausa `amr` berikut:

```
"ForAnyValue:StringLike": {  
  "cognito-identity.amazonaws.com:amr": "graph.facebook.com"  
}
```

## Kebijakan akses

Izin yang Anda lampirkan ke peran berlaku untuk semua pengguna yang mengambil peran tersebut. Untuk mempartisi akses pengguna Anda, gunakan kondisi kebijakan dan variabel. Untuk informasi selengkapnya, lihat [elemen kebijakan IAM: Variabel dan tag](#). Anda dapat menggunakan sub kondisi untuk membatasi tindakan ke ID identitas Amazon Cognito dalam kebijakan akses Anda. Gunakan opsi ini dengan hati-hati, terutama untuk identitas yang tidak diautentikasi, yang tidak memiliki ID pengguna yang konsisten. Untuk informasi selengkapnya tentang variabel kebijakan IAM untuk federasi web dengan Amazon Cognito, [lihat IAM AWS STS dan kunci konteks kondisi](#) di AWS Identity and Access Management Panduan Pengguna.

[Untuk perlindungan keamanan tambahan, Amazon Cognito menerapkan kebijakan penurunan cakupan ke kredensial yang Anda tetapkan kepada pengguna yang tidak diautentikasi dalam alur yang disempurnakan](#). `GetCredentialsForIdentity` Kebijakan cakupan bawah menambahkan [Kebijakan sesi inline](#) dan ke kebijakan IAM yang Anda [AWS kebijakan sesi terkelola](#) terapkan pada peran Anda yang tidak diautentikasi. Karena Anda harus memberikan akses baik dalam kebijakan IAM untuk peran Anda dan kebijakan sesi, kebijakan cakupan bawah membatasi akses pengguna ke layanan selain yang ada dalam daftar berikut.

### Note

Dalam alur dasar (klasik), Anda membuat permintaan [AssumeRoleWithWebIdentity](#) API sendiri, dan dapat menerapkan batasan ini pada permintaan. Sebagai praktik keamanan terbaik, jangan tetapkan izin apa pun di atas kebijakan cakupan bawah ini kepada pengguna yang tidak diautentikasi.

Amazon Cognito juga mencegah pengguna yang diautentikasi dan tidak diautentikasi membuat permintaan API ke kumpulan identitas Amazon Cognito dan Sinkronisasi Amazon Cognito. Lain Layanan AWS mungkin menempatkan pembatasan akses layanan dari identitas web.

Dalam permintaan yang berhasil dengan alur yang disempurnakan, Amazon Cognito membuat permintaan `AssumeRoleWithWebIdentity` API di latar belakang. Di antara parameter dalam permintaan ini, Amazon Cognito menyertakan yang berikut ini.

1. ID identitas pengguna Anda.
2. ARN dari peran IAM yang ingin diasumsikan oleh pengguna Anda.
3. `policyParameter` yang menambahkan kebijakan sesi inline.
4. `PolicyArns.member.NParameter` yang nilainya adalah kebijakan AWS terkelola yang memberikan izin tambahan di Amazon. CloudWatch

Layanan yang dapat diakses oleh pengguna yang tidak diautentikasi

Saat Anda menggunakan alur yang disempurnakan, kebijakan penurun cakupan yang diterapkan Amazon Cognito pada sesi pengguna mencegah mereka menggunakan layanan apa pun selain yang tercantum dalam tabel berikut. Untuk subset layanan, hanya tindakan spesifik yang diizinkan.

Kategori	Layanan
Analitik	Amazon Data Firehose
	Layanan Terkelola Amazon untuk Apache Flink
Integrasi Aplikasi	Amazon Simple Queue Service
AR & VR	Amazon Sumeria <sup>1</sup>
Aplikasi Bisnis	Amazon Mobile Analytics
	Layanan Email Sederhana Amazon
Hitung	AWS Lambda
Kriptografi & PKI	AWS Key Management Service <sup>1</sup>
Basis Data	Amazon DynamoDB
	Amazon SimpleDB
Web & Seluler Front-end	AWS AppSync
	Amazon Location Service
	Amazon Simple Notification Service



Kategori	Layanan
	Amazon Pinpoint
Pengembangan Game	Amazon GameLift
Internet of Things (IoT)	AWS IoT
Machine Learning	Amazon CodeWhisperer Amazon Comprehend Amazon Lex Amazon Machine Learning Amazon Personalize Amazon Polly Amazon Rekognition Amazon SageMaker <sup>1</sup> Amazon Textract <sup>1</sup> Amazon Transcribe Amazon Translate
Manajemen & Tata Kelola	Amazon CloudWatch CloudWatch Log Amazon
Jaringan & Pengiriman Konten	Amazon API Gateway
Keamanan, Identitas, & Kepatuhan	Kolam pengguna Amazon Cognito
Penyimpanan	Amazon Simple Storage Service

<sup>1</sup> Layanan AWS Untuk tabel berikut, kebijakan inline memberikan subset tindakan. Tabel menampilkan tindakan yang tersedia di masing-masing.

Layanan AWS	Izin maksimum untuk pengguna aliran yang disempurnakan yang tidak diautentikasi
AWS Key Management Service	Encrypt Decrypt ReEncrypt GenerateDataKey
Amazon SageMaker	InvokeEndpoint
Amazon Textract	DetectDocumentText AnalyzeDocument
Amazon Sumerian	View*

Untuk memberikan akses ke Layanan AWS luar daftar ini, aktifkan alur otentikasi dasar (klasik) di kumpulan identitas Anda. Jika pengguna Anda melihat `NotAuthorizedException` kesalahan Layanan AWS yang diizinkan oleh kebijakan yang ditetapkan ke peran IAM untuk pengguna yang tidak diautentikasi, evaluasi apakah Anda dapat menghapus layanan tersebut dari kasus penggunaan Anda. Jika Anda tidak bisa, beralihlah ke aliran dasar.

### Kebijakan sesi inline

Kebijakan sesi inline membatasi izin efektif pengguna Anda untuk menyertakan akses ke Layanan AWS bagian luar yang ada dalam daftar berikut. Anda juga harus memberikan izin untuk ini Layanan AWS dalam kebijakan yang Anda terapkan pada peran IAM pengguna. Izin efektif pengguna untuk sesi peran yang diasumsikan adalah persimpangan kebijakan yang ditetapkan untuk peran mereka, dan kebijakan sesi mereka. Untuk informasi selengkapnya, lihat [Kebijakan sesi](#) di Panduan AWS Identity and Access Management Pengguna.

Amazon Cognito menambahkan kebijakan sebaris berikut ke sesi untuk pengguna Anda Wilayah AWS yang diaktifkan secara default.

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

```
{
  "Effect": "Allow",
  "Action": [
    "cloudwatch:*",
    "logs:*",
    "dynamodb:*",
    "kinesis:*",
    "mobileanalytics:*",
    "s3:*",
    "ses:*",
    "sns:*",
    "sqs:*",
    "lambda:*",
    "machinelearning:*",
    "execute-api:*",
    "iot:*",
    "gamelift:*",
    "scs:*",
    "cognito-identity:*",
    "cognito-idp:*",
    "lex:*",
    "polly:*",
    "comprehend:*",
    "translate:*",
    "transcribe:*",
    "rekognition:*",
    "mobiletargeting:*",
    "firehose:*",
    "appsync:*",
    "personalize:*",
    "kms:Encrypt",
    "kms:Decrypt",
    "kms:ReEncrypt*",
    "kms:GenerateDataKey*",
    "sagemaker:InvokeEndpoint",
    "cognito-sync:*",
    "sumerian:View*",
    "codewhisperer:*",
    "textextract:DetectDocumentText",
    "textextract:AnalyzeDocument",
    "sdb:*"
  ],
  "Resource": [
    "*"
  ]
}
```

```

    ]
  }
]
}

```

Untuk semua Wilayah lainnya, kebijakan inline scope-down mencakup semua yang tercantum di Wilayah default kecuali untuk pernyataan berikut. Action

```

    "cognito-sync:*",
    "sumerian:View*",
    "codewhisperer:*",
    "textract:DetectDocumentText",
    "textract:AnalyzeDocument",
    "sdb:*"

```

### Kebijakan sesi AWS terkelola

Amazon Cognito juga membatasi cakupan izin pengguna yang tidak diautentikasi dengan kebijakan AWS terkelola `AmazonCognitoUnAuthedIdentitiesSessionPolicy` untuk pengguna Anda yang tidak diautentikasi dalam alur yang disempurnakan. Anda juga harus memberikan izin ini dalam kebijakan yang Anda lampirkan ke peran IAM Anda yang tidak diautentikasi.

Kebijakan `AmazonCognitoUnAuthedIdentitiesSessionPolicy` terkelola memiliki izin berikut.

```

{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": [
      "rum:PutRumEvents",
      "polly:*",
      "comprehend:*",
      "translate:*",
      "transcribe:*",
      "rekognition:*",
      "mobiletargeting:*",
      "firehose:*",
      "personalize:*",
      "sagemaker:InvokeEndpoint"
    ],
    "Resource": "*"
  }]
}

```

```
}
```

## Contoh kebijakan akses

Di bagian ini, Anda dapat menemukan contoh kebijakan akses Amazon Cognito yang memberi pengguna izin minimum yang diperlukan untuk melakukan operasi tertentu. Anda dapat lebih membatasi izin untuk ID identitas yang diberikan dengan menggunakan variabel kebijakan jika memungkinkan. Misalnya, menggunakan `${cognito-identity.amazonaws.com:sub}`. Untuk informasi selengkapnya, lihat: [Memahami Autentikasi Amazon Cognito Bagian 3: Peran dan Kebijakan](#) pada Blog Seluler AWS .

### Note

Sebagai praktik terbaik keamanan, kebijakan harus menyertakan hanya izin yang diperlukan pengguna untuk melakukan tugas-tugas mereka. Ini berarti bahwa Anda harus mencoba untuk selalu menjangkau akses ke identitas individu untuk objek bila memungkinkan.

## Berikan akses baca identitas ke satu objek di Amazon S3

Kebijakan akses berikut memberikan izin baca ke identitas untuk mengambil objek tunggal dari bucket S3 yang diberikan.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "s3:GetObject"
      ],
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::mybucket/assets/my_picture.jpg"]
    }
  ]
}
```

## Berikan identitas akses baca dan tulis ke jalur khusus identitas di Amazon S3

Kebijakan akses berikut memberikan izin baca dan tulis untuk mengakses “folder” prefiks tertentu di bucket S3 dengan memetakan awalan ke variabel `${cognito-identity.amazonaws.com:sub}`.

Dengan kebijakan ini, identitas seperti yang us-east-1:12345678-1234-1234-1234-123456790ab disisipkan via `${cognito-identity.amazonaws.com:sub}` bisa mendapatkan, menempatkan, dan mencantumkan objek ke dalam `arn:aws:s3:::mybucket/us-east-1:12345678-1234-1234-1234-123456790ab`. Namun, identitas tidak akan diberikan akses ke objek lain di `arn:aws:s3:::mybucket`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": ["s3:ListBucket"],
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::mybucket"],
      "Condition": {"StringLike": {"s3:prefix": ["${cognito-identity.amazonaws.com:sub}/*"]}}
    },
    {
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::mybucket/${cognito-identity.amazonaws.com:sub}/*"]
    }
  ]
}
```

Tetapkan identitas akses berbutir halus ke Amazon DynamoDB

Kebijakan akses berikut menyediakan kontrol akses berbutir halus ke sumber daya DynamoDB menggunakan variabel lingkungan Amazon Cognito. Variabel ini memberikan akses ke item di DynamoDB dengan ID identitas. Untuk informasi selengkapnya, lihat [Menggunakan Kondisi Kebijakan IAM untuk Kontrol Akses Mendetail](#) di Panduan Developer Amazon DynamoDB.

```
{
  "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:us-west-2:123456789012:table/MyTable"
  ],
  "Condition": {
    "ForAllValues:StringEquals": {
      "dynamodb:LeadingKeys": ["${cognito-identity.amazonaws.com:sub}"]
    }
  }
}
]
}

```

Berikan izin identitas untuk menjalankan fungsi Lambda

Kebijakan akses berikut memberikan izin identitas untuk menjalankan fungsi Lambda.

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "lambda:InvokeFunction",
      "Resource": [
        "arn:aws:lambda:us-west-2:123456789012:function:MyFunction"
      ]
    }
  ]
}

```

Berikan izin identitas untuk mempublikasikan catatan ke Kinesis Data Streams

Kebijakan akses berikut mengizinkan identitas untuk menggunakan operasi PutRecord dengan salah satu Kinesis Data Streams. Hal ini dapat diterapkan pada pengguna yang perlu menambahkan catatan data ke semua aliran di akun. Untuk informasi lebih lanjut, lihat [Mengontrol Akses ke Sumber Daya Amazon Kinesis Data Streams Menggunakan IAM](#) di Panduan Developer Amazon Kinesis Data Streams.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "kinesis:PutRecord",
      "Resource": [
        "arn:aws:kinesis:us-east-1:111122223333:stream/stream1"
      ]
    }
  ]
}
```

Berikan akses identitas ke data mereka di toko Amazon Cognito Sync

Kebijakan akses berikut memberikan izin identitas untuk hanya mengakses data mereka sendiri di toko Sinkronisasi Amazon Cognito.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": "cognito-sync:*",
    "Resource": ["arn:aws:cognito-sync:us-east-1:123456789012:identitypool/${cognito-identity.amazonaws.com:aud}/identity/${cognito-identity.amazonaws.com:sub}/*"]
  }]
}
```

## Kepercayaan peran dan izin

Cara peran ini berbeda adalah dalam hubungan kepercayaan mereka. Berikut ini adalah contoh kebijakan kepercayaan untuk peran yang tidak diautentikasi:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Federated": "cognito-identity.amazonaws.com"
      }
    }
  ]
}
```



```

    },
    "Action": "sts:AssumeRoleWithWebIdentity",
    "Condition": {
      "StringEquals": {
        "cognito-identity.amazonaws.com:aud": "us-east-1:12345678-corner-
cafe-123456790ab"
      },
      "ForAnyValue:StringLike": {
        "cognito-identity.amazonaws.com:amr": "unauthenticated"
      }
    }
  }
]
}

```

Kebijakan ini memberikan izin kepada pengguna federasi dari `cognito-identity.amazonaws.com` (penerbit token OpenID Connect) untuk mengambil peran ini. Selain itu, kebijakan membatasi `aud` pada token, dalam hal ini ID kolom identitas, untuk mencocokkan kolom identitas tersebut. Terakhir, kebijakan menetapkan bahwa salah satu anggota array dari `amr` klaim multi-nilai token yang dikeluarkan oleh operasi Amazon `GetOpenIdToken` Cognito API memiliki nilai `unauthenticated`

Saat Amazon Cognito membuat token, ia menetapkan token sebagai salah satu `atauunauthenticated`. `amr` `authenticated` Jika `amr` `yaauthenticated`, token mencakup penyedia apa pun yang digunakan selama otentikasi. Ini berarti Anda dapat membuat peran yang hanya mempercayai pengguna yang masuk melalui Facebook dengan mengubah `amr` kondisi seperti yang ditunjukkan:

```

"ForAnyValue:StringLike": {
  "cognito-identity.amazonaws.com:amr": "graph.facebook.com"
}

```

Berhati-hatilah saat mengubah hubungan kepercayaan Anda pada peran Anda, atau saat mencoba menggunakan peran di seluruh kolom identitas. Jika Anda tidak mengonfigurasi peran Anda dengan benar untuk mempercayai kumpulan identitas Anda, pengecualian dari hasil STS, seperti berikut ini:

```

AccessDenied -- Not authorized to perform sts:AssumeRoleWithWebIdentity

```

Jika Anda melihat pesan ini, periksa apakah kumpulan identitas dan jenis otentikasi Anda memiliki peran yang sesuai.

# Praktik terbaik keamanan untuk kumpulan identitas Amazon Cognito

Kumpulan identitas Amazon Cognito menyediakan AWS kredensial sementara untuk aplikasi Anda. Akun AWS sering berisi sumber daya yang dibutuhkan pengguna aplikasi Anda, dan sumber daya back-end pribadi. Peran dan kebijakan IAM yang membentuk AWS kredensial dapat memberikan akses ke salah satu sumber daya ini.

Praktik terbaik utama konfigurasi kumpulan identitas adalah memastikan bahwa aplikasi Anda dapat menyelesaikan pekerjaan tanpa kelebihan atau hak istimewa yang tidak diinginkan. Untuk mencegah kesalahan konfigurasi keamanan, tinjau rekomendasi ini sebelum peluncuran setiap aplikasi yang ingin Anda rilis ke produksi.

## Topik

- [Praktik terbaik konfigurasi IAM](#)
- [Praktik terbaik konfigurasi kumpulan identitas](#)

## Praktik terbaik konfigurasi IAM

Saat tamu atau pengguna yang diautentikasi memulai sesi di aplikasi Anda yang memerlukan kredensial kumpulan identitas, aplikasi Anda akan mengambil kredensial sementara AWS untuk peran IAM. Kredensialnya mungkin untuk peran default, peran yang dipilih berdasarkan aturan dalam konfigurasi kumpulan identitas, atau untuk peran khusus yang dipilih oleh aplikasi Anda. Dengan izin yang ditetapkan untuk setiap peran, pengguna Anda mendapatkan akses ke AWS sumber daya Anda.

Untuk informasi selengkapnya tentang praktik terbaik IAM umum, lihat [praktik terbaik IAM](#) di AWS Identity and Access Management Panduan Pengguna.

## Gunakan kondisi kebijakan kepercayaan dalam peran IAM

IAM mensyaratkan bahwa peran untuk kumpulan identitas memiliki setidaknya satu kondisi kebijakan kepercayaan. Kondisi ini dapat, misalnya, mengatur cakupan peran ke pengguna yang diautentikasi saja. AWS STS juga mensyaratkan bahwa permintaan otentikasi dasar lintas akun memiliki dua kondisi khusus: `cognito-identity.amazonaws.com:aud` dan `cognito-identity.amazonaws.com:amr`. Sebagai praktik terbaik, terapkan kedua kondisi ini di semua

peran IAM yang mempercayai prinsip `cognito-identity.amazonaws.com` layanan kumpulan identitas.

- `cognito-identity.amazonaws.com:aud`: Klaim `aud` dalam token kumpulan identitas harus cocok dengan ID kumpulan identitas tepercaya.
- `cognito-identity.amazonaws.com:amr`: Klaim `amr` dalam token kumpulan identitas harus diautentikasi atau tidak diautentikasi. Dengan kondisi ini, Anda dapat memesan akses ke peran hanya untuk tamu yang tidak diautentikasi, atau hanya untuk pengguna yang diautentikasi. Anda dapat menyempurnakan nilai kondisi ini lebih lanjut untuk membatasi peran kepada pengguna dari penyedia tertentu, misalnya `graph.facebook.com`

Contoh kebijakan kepercayaan peran berikut memberikan akses ke peran dalam kondisi berikut:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Federated": "cognito-identity.amazonaws.com"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "cognito-identity.amazonaws.com:aud": "us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
        },
        "ForAnyValue:StringLike": {
          "cognito-identity.amazonaws.com:amr": "authenticated"
        }
      }
    }
  ]
}
```

Elemen yang berhubungan dengan kumpulan identitas

- `"Federated": "cognito-identity.amazonaws.com"`: Pengguna harus berasal dari kumpulan identitas.

- "cognito-identity.amazonaws.com:aud": "us-east-1:a1b2c3d4-5678-90ab-cdef-example11111": Pengguna harus berasal dari kumpulan identitas tertentu - east-1:a1b2c3d4-5678-90ab-cdef-example11111.
- "cognito-identity.amazonaws.com:amr": "authenticated": Pengguna harus diautentikasi. Pengguna tamu tidak dapat mengambil peran tersebut.

## Terapkan izin hak istimewa paling sedikit

Saat Anda menetapkan izin dengan kebijakan IAM untuk akses terautentikasi atau akses tamu, berikan hanya izin khusus yang diperlukan untuk melakukan tugas tertentu, atau paling tidak izin hak istimewa. Contoh kebijakan IAM berikut, bila diterapkan ke peran, memberikan akses hanya-baca ke satu file gambar di bucket Amazon S3.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "s3:GetObject"
      ],
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::mybucket/assets/my_picture.jpg"]
    }
  ]
}
```

## Praktik terbaik konfigurasi kumpulan identitas

Identity pool memiliki opsi fleksibel untuk menghasilkan AWS kredensi. Jangan mengambil pintasan desain ketika aplikasi Anda dapat bekerja dengan metode yang paling aman.

### Memahami efek dari akses tamu

Akses tamu yang tidak diautentikasi memungkinkan pengguna untuk mengambil data dari Anda Akun AWS sebelum mereka masuk. Siapa pun yang mengetahui ID kumpulan identitas Anda dapat meminta kredensial yang tidak diautentikasi. ID kumpulan identitas Anda bukanlah informasi rahasia. Saat Anda mengaktifkan akses tamu, AWS izin yang Anda berikan untuk sesi yang tidak diautentikasi tersedia untuk semua orang.

Sebagai praktik terbaik, biarkan akses tamu dinonaktifkan dan ambil sumber daya yang diperlukan hanya setelah pengguna mengautentikasi. Jika aplikasi Anda memerlukan akses ke sumber daya sebelum masuk, lakukan tindakan pencegahan berikut.

- Biasakan diri Anda dengan [batasan otomatis yang ditempatkan pada peran yang tidak diautentikasi](#).
- Pantau dan sesuaikan izin peran IAM Anda yang tidak diautentikasi agar sesuai dengan kebutuhan spesifik aplikasi Anda.
- Berikan akses ke sumber daya tertentu.
- Amankan kebijakan kepercayaan peran IAM default yang tidak diautentikasi.
- Aktifkan akses tamu hanya jika Anda yakin bahwa Anda akan memberikan izin dalam peran IAM Anda kepada siapa pun di internet.

## Gunakan otentikasi yang disempurnakan secara default

Dengan autentikasi dasar (klasik), Amazon Cognito mendelegasikan pemilihan peran IAM ke aplikasi Anda. Sebaliknya, aliran yang ditingkatkan menggunakan logika terpusat di kumpulan identitas Anda untuk menentukan peran IAM. Ini juga menyediakan keamanan tambahan untuk identitas yang tidak diautentikasi dengan [kebijakan cakupan bawah yang](#) menetapkan batas atas izin IAM. Aliran yang ditingkatkan adalah pilihan paling aman dengan tingkat upaya pengembang terendah. Untuk mempelajari lebih lanjut tentang opsi ini, lihat [Aliran otentikasi kumpulan identitas \(identitas federasi\)](#).

Alur dasar dapat mengekspos logika sisi klien yang masuk ke pemilihan peran dan perakitan permintaan AWS STS API untuk kredensial. Alur yang disempurnakan menyembunyikan logika dan permintaan peran asumsi di balik otomatisasi kumpulan identitas.

Saat Anda mengonfigurasi otentikasi dasar, terapkan [praktik terbaik IAM](#) ke peran IAM dan izinnya.

## Gunakan penyedia pengembang dengan aman

Identitas otentikasi pengembang adalah fitur kumpulan identitas untuk aplikasi sisi server. Satu-satunya bukti otentikasi yang diperlukan kumpulan identitas untuk otentikasi pengembang adalah AWS kredensial pengembang kumpulan identitas. Kumpulan identitas tidak memberlakukan batasan apa pun pada validitas pengidentifikasi penyedia pengembang yang Anda tampilkan dalam alur autentikasi ini.

Sebagai praktik terbaik, hanya menerapkan penyedia pengembang dalam kondisi berikut:

- Untuk membuat akuntabilitas atas penggunaan kredensial yang diautentikasi pengembang, rancang nama dan pengidentifikasi penyedia pengembang Anda untuk menunjukkan sumber otentikasi. Misalnya: "Logins" : {"MyCorp provider" : "[*provider application ID*]"}.
- Hindari kredensial pengguna yang berumur panjang. [Konfigurasi klien sisi server Anda untuk meminta identitas dengan peran terkait layanan seperti profil instans EC2 dan peran eksekusi Lambda.](#)
- Hindari pencampuran sumber kepercayaan internal dan eksternal dalam kumpulan identitas yang sama. Tambahkan penyedia pengembang Anda dan penyedia sistem masuk tunggal (SSO) Anda di kumpulan identitas terpisah.

## Menggunakan atribut untuk kontrol akses

Atribut untuk kontrol akses adalah implementasi kumpulan identitas Amazon Cognito dari kontrol akses berbasis atribut (ABAC). Anda dapat menggunakan kebijakan IAM untuk mengontrol akses ke sumber daya AWS melalui kolam identitas Amazon Cognito berdasarkan atribut pengguna. Atribut ini dapat diambil dari penyedia identitas sosial dan perusahaan. Anda dapat memetakan atribut dalam akses penyedia dan token ID atau pernyataan SAML ke tanda yang dapat direferensikan dalam kebijakan izin IAM.

Anda dapat memilih pemetaan default atau membuat pemetaan kustom Anda sendiri di kolam identitas Amazon Cognito. Pemetaan default memungkinkan Anda untuk menulis kebijakan IAM berdasarkan set tetap atribut pengguna. Pemetaan kustom memungkinkan Anda untuk memilih serangkaian kustom atribut pengguna yang direferensikan dalam kebijakan izin IAM. Nama atribut di konsol Amazon Cognito dipetakan ke Kunci tanda untuk utama, yang merupakan tanda yang direferensikan dalam kebijakan izin IAM.

Misalnya, katakanlah Anda memiliki layanan streaming media dengan keanggotaan gratis dan berbayar. Anda menyimpan file media di Amazon S3 dan menandai file tersebut dengan tanda gratis atau premium. Anda dapat menggunakan atribut untuk kontrol akses untuk memungkinkan akses ke konten gratis dan berbayar berdasarkan tingkat keanggotaan pengguna, yang merupakan bagian dari profil pengguna. Anda dapat memetakan atribut keanggotaan untuk kunci tanda yang utama untuk diteruskan ke kebijakan izin IAM. Dengan cara ini Anda dapat membuat kebijakan izin tunggal dan secara kondisional mengizinkan akses ke konten premium berdasarkan nilai tingkat keanggotaan dan tanda pada file konten.

### Topik

- [Menggunakan atribut untuk kontrol akses dengan kolam identitas Amazon Cognito](#)

- [Menggunakan atribut untuk contoh kebijakan kontrol akses](#)
- [Matikan atribut untuk kontrol akses \(konsol\)](#)
- [Pemetaan penyedia default](#)

Menggunakan atribut untuk kontrol akses memiliki beberapa manfaat:

- Manajemen izin lebih efisien ketika Anda menggunakan atribut untuk kontrol akses. Anda dapat membuat kebijakan izin dasar yang menggunakan atribut pengguna bukan menciptakan beberapa kebijakan untuk fungsi pekerjaan yang berbeda.
- Anda tidak perlu memperbarui kebijakan kapan pun Anda menambahkan atau menghapus sumber daya atau pengguna untuk aplikasi Anda. Kebijakan izin hanya akan memberikan akses ke pengguna dengan atribut pengguna yang cocok. Misalnya, Anda mungkin perlu mengontrol akses ke bucket S3 tertentu berdasarkan judul pekerjaan pengguna. Dalam hal ini, Anda dapat membuat kebijakan izin untuk mengizinkan akses ke file ini hanya untuk pengguna dalam jabatan pekerjaan yang ditentukan. Untuk informasi selengkapnya, lihat [Tutorial IAM: Menggunakan tanda sesi SAML untuk ABAC](#).
- Atribut dapat dilewatkan sebagai tanda utama untuk kebijakan yang memungkinkan atau menolak izin berdasarkan nilai-nilai atribut tersebut.

## Menggunakan atribut untuk kontrol akses dengan kolam identitas Amazon Cognito

Sebelum Anda dapat menggunakan atribut untuk kontrol akses, pastikan bahwa Anda memenuhi prasyarat berikut:

- [AWS Akun](#)
- [Kolam pengguna](#)
- [Kolam identitas](#)
- [Siapkan SDK](#)
- [Penyedia identitas terintegrasi](#)
- [Kredensialnya](#)

Untuk menggunakan atribut untuk kontrol akses, Klaim yang Anda tetapkan sebagai sumber data menetapkan nilai Kunci Tag yang Anda pilih. Amazon Cognito menerapkan kunci tag dan nilai

ke sesi pengguna Anda. Kebijakan IAM Anda dapat mengevaluasi akses pengguna Anda dari `{aws:PrincipalTag/tagkey}` kondisi tersebut. IAM mengevaluasi nilai tag pengguna Anda terhadap kebijakan.

Anda harus menyiapkan peran IAM yang kredensialnya ingin Anda berikan kepada pengguna Anda. Kebijakan kepercayaan dari peran ini harus mengizinkan Amazon Cognito untuk mengambil peran bagi pengguna Anda. Untuk atribut untuk kontrol akses, Anda juga harus mengizinkan Amazon Cognito menerapkan tag utama ke sesi sementara pengguna Anda. Berikan izin untuk mengambil peran dengan tindakan tersebut [AssumeRoleWithWebIdentity](#). Berikan izin untuk menandai sesi pengguna dengan tindakan khusus [izin sts:TagSession](#) Untuk informasi selengkapnya, lihat [Melewati tag sesi AWS Security Token Service di Panduan AWS Identity and Access Management Pengguna](#). Untuk contoh kebijakan kepercayaan yang memberikan `sts:AssumeRoleWithWebIdentity` dan `sts:TagSession` izin kepada prinsipal layanan `cognito-identity.amazonaws.com` Amazon Cognito, lihat. [Menggunakan atribut untuk contoh kebijakan kontrol akses](#)

Untuk mengkonfigurasi atribut untuk kontrol akses di konsol

1. Masuk ke [konsol Amazon Cognito](#) dan pilih Identity pool. Pilih kumpulan identitas.
2. Pilih tab Akses pengguna.
3. Temukan penyedia Identitas. Pilih penyedia identitas yang ingin Anda edit. Jika Anda ingin menambahkan IDP baru, pilih Tambahkan penyedia identitas.
4. Untuk mengubah tag utama yang ditetapkan Amazon Cognito saat mengeluarkan kredensial kepada pengguna yang telah diautentikasi dengan penyedia ini, pilih Edit di Atribut untuk kontrol akses.
  - a. Untuk tidak menerapkan tag utama, pilih Tidak aktif.
  - b. Untuk menerapkan tag utama berdasarkan sub dan aud klaim, pilih Gunakan pemetaan default.
  - c. Untuk membuat skema atribut kustom Anda sendiri ke tag utama, pilih Gunakan pemetaan khusus. Kemudian masukkan kunci Tag yang ingin Anda sumber dari setiap Klaim yang ingin Anda wakili dalam tag.
5. Pilih Simpan perubahan.



## Menggunakan atribut untuk contoh kebijakan kontrol akses

Pertimbangkan skenario di mana seorang karyawan dari departemen hukum perusahaan perlu membuat daftar semua file dalam bucket milik departemen mereka dan diklasifikasikan dengan tingkat keamanan mereka. Asumsikan token bahwa karyawan ini mendapat dari penyedia identitas berisi klaim berikut.

### Klaim

```
{ .
  .
  "sub" : "57e7b692-4f66-480d-98b8-45a6729b4c88",
  "department" : "legal",
  "clearance" : "confidential",
  .
  .
}
```

Atribut ini dapat dipetakan ke tanda dan direferensikan dalam kebijakan izin IAM sebagai tanda utama. Anda sekarang dapat mengelola akses dengan mengubah profil pengguna di ujung penyedia identitas. Atau, Anda dapat mengubah atribut pada sisi sumber daya dengan menggunakan nama atau tanda tanpa mengubah kebijakan itu sendiri.

Kebijakan izin berikut melakukan dua hal:

- Mengizinkan akses daftar ke semua bucket S3 yang diakhiri dengan awalan yang cocok dengan nama departemen pengguna.
- Mengizinkan akses baca pada file dalam bucket ini selama tanda izin pada file cocok dengan atribut izin pengguna.

### Kebijakan izin

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

```

    "Action": "s3:List*",
    "Resource": "arn:aws:s3::*-${aws:PrincipalTag/department}"
  },
  {
    "Effect": "Allow",
    "Action": "s3:GetObject*",
    "Resource": "arn:aws:s3::*-${aws:PrincipalTag/department}/*",
    "Condition": {
      "StringEquals": {
        "s3:ExistingObjectTag/clearance": "${aws:PrincipalTag/clearance}"
      }
    }
  }
]
}

```

Kebijakan kepercayaan menentukan siapa yang dapat mengambil peran ini. Kebijakan hubungan kepercayaan memungkinkan penggunaan `sts:AssumeRoleWithWebIdentity` dan `sts:TagSession` untuk mengizinkan akses. Ini menambahkan kondisi untuk membatasi kebijakan ke kumpulan identitas yang Anda buat dan memastikan bahwa itu untuk peran yang diautentikasi.

### Kebijakan kepercayaan

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "cognito-identity.amazonaws.com"
      },
      "Action": [
        "sts:AssumeRoleWithWebIdentity",
        "sts:TagSession"
      ],
      "Condition": {
        "StringEquals": {
          "cognito-identity.amazonaws.com:aud": "IDENTITY-POOL-ID"
        },
        "ForAnyValue:StringLike": {

```

```

    "cognito-identity.amazonaws.com:amr": "authenticated"
  }
}
]
}

```

## Matikan atribut untuk kontrol akses (konsol)

Ikuti prosedur ini untuk menonaktifkan atribut untuk kontrol akses.

Untuk menonaktifkan atribut untuk kontrol akses di konsol

1. Masuk ke [konsol Amazon Cognito](#) dan pilih Identity pool. Pilih kumpulan identitas.
2. Pilih tab Akses pengguna.
3. Temukan penyedia Identitas. Pilih penyedia identitas yang ingin Anda edit.
4. Pilih Edit di Atribut untuk kontrol akses.
5. Untuk tidak menerapkan tag utama, pilih Tidak aktif.
6. Pilih Simpan perubahan.

## Pemetaan penyedia default

Tabel berikut memiliki informasi pemetaan default untuk penyedia autentikasi yang didukung Amazon Cognito.

Penyedia	Jenis token	Nilai tanda utama	Contoh
Kolam pengguna Amazon Cognito	Token ID	aud (ID klien) dan sub (ID pengguna)	"6jk8ltokc7ac9es6j rtg9q572f", "57e7b692 -4f66-480d-98b8-45 a6729b4c88"
Facebook	Token akses	aud(app_id), sub(user_id)	"492844718097981", "112177216992379"
Google	Token ID	aud (ID klien) dan sub (ID pengguna)	"620493171733-eebk 7c0hcp5lj3e1tlqp1g

Penyedia	Jenis token	Nilai tanda utama	Contoh
			ntt3k0rncv.apps.googleusercontent.com", "109220063452404746097"
SAML	Pernyataan	"http://schemas.xmlsoap.org/ws/2005/05/identity/claims/nameidentifier" , "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/name"	"auth0 5e28d196f8f55a0eaaa95de3", user123@gmail.com"
Apple	Token ID	aud (ID klien) dan sub (ID pengguna)	"com.amazonaws.ec2-54-80-172-243.compute-1.client", "001968.a6ca34e9c1e742458a26cf8005854be9.0733"
Amazon	Token akses	aud (ID klien di Amzn Dev Ac), user_id(ID pengguna)	"amzn1.application-oa2-client.9d70d9382d3446108aaee3dd763a0fa6", "amzn1.account.AGHNIFJQMFSBG3G6XCPVB35ORQAA"
Penyedia OIDC Standar	ID dan token akses	aud (sebagai client_id), sub (sebagai ID pengguna)	"620493171733-eebk7c0hcp5lj3e1tlqp1gntt3k0rncv.apps.googleusercontent.com", "109220063452404746097"

Penyedia	Jenis token	Nilai tanda utama	Contoh
Twitter	Token akses	aud (ID aplikasi; aplikasi Rahasia), sub (ID pengguna)	"DfwifTtKex1fiibrn otlr0cfk; xgj5xb8xi rivc lidkw7fxmwcj k9 1y5z1", "12690038 84292222976" PjXg JrFvno gwZkLexo
DevAuth	Peta	Tidak berlaku	"tag1", "tag2"

### Note

Opsi pemetaan atribut default secara otomatis diisi untuk Kunci Tanda untuk Utama dan nama Atribut. Anda tidak dapat mengubah pemetaan default.

## Menggunakan kontrol akses berbasis peran

Kumpulan identitas Amazon Cognito menetapkan pengguna Anda yang diautentikasi satu set kredensial hak istimewa terbatas sementara untuk mengakses sumber daya Anda. AWS Izin untuk setiap pengguna dikendalikan melalui [IAM role](#) yang Anda buat. Anda dapat menentukan aturan untuk memilih peran bagi setiap pengguna berdasarkan klaim di token ID pengguna. Anda dapat menentukan peran default untuk pengguna terautentikasi. Anda juga dapat menentukan IAM role terpisah dengan izin terbatas untuk pengguna tamu yang tidak terautentikasi.

## Membuat peran untuk pemetaan peran

Penting untuk menambahkan kebijakan kepercayaan yang tepat untuk setiap peran sehingga peran tersebut hanya dapat diasumsikan oleh Amazon Cognito untuk pengguna terautentikasi dalam kolom identitas Anda. Berikut adalah contoh kebijakan kepercayaan tersebut:

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

```

    "Effect": "Allow",
    "Principal": {
      "Federated": "cognito-identity.amazonaws.com"
    },
    "Action": "sts:AssumeRoleWithWebIdentity",
    "Condition": {
      "StringEquals": {
        "cognito-identity.amazonaws.com:aud": "us-east-1:12345678-corner-
cafe-123456790ab"
      },
      "ForAnyValue:StringLike": {
        "cognito-identity.amazonaws.com:amr": "authenticated"
      }
    }
  }
]
}

```

Kebijakan ini mengizinkan pengguna gabungan dari `cognito-identity.amazonaws.com` (penerbit token OpenID Connect) untuk mengambil peran ini. Selain itu, kebijakan membatasi `aud` pada token, dalam hal ini ID kolam identitas, untuk mencocokkan kolam identitas tersebut. Terakhir, kebijakan menetapkan bahwa salah satu anggota array dari `amr` klaim multi-nilai token yang dikeluarkan oleh tindakan Amazon `GetOpenIdToken` Cognito API memiliki nilai `authenticated`

## Memberikan izin peran masuk

Untuk memungkinkan pengguna menyetel peran dengan izin melebihi izin pengguna yang ada di kumpulan identitas, beri mereka `iam:PassRole` izin untuk meneruskan peran tersebut set `identity-pool-roles` ke API. Sebagai contoh, jika pengguna tidak dapat menulis ke Amazon S3, tetapi IAM role yang ditetapkan pengguna pada kolam identitas memberikan izin menulis ke Amazon S3, pengguna hanya dapat mengatur peran ini jika izin `iam:PassRole` diberikan untuk peran tersebut. Contoh kebijakan berikut menunjukkan cara memberikan izin `iam:PassRole`.

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Stmt1",
      "Effect": "Allow",
      "Action": [
        "iam:PassRole"
      ]
    }
  ]
}

```

```
    ],
    "Resource": [
      "arn:aws:iam::123456789012:role/myS3WriteAccessRole"
    ]
  }
]
```

Dalam contoh kebijakan ini, izin `iam:PassRole` diberikan untuk peran `myS3WriteAccessRole`. Peran ditentukan menggunakan Amazon Resource Name (ARN) peran. Anda juga harus melampirkan kebijakan ini kepada pengguna Anda. Untuk informasi lebih lanjut, lihat [Bekerja dengan Kebijakan Terkelola](#).

#### Note

Fungsi Lambda menggunakan kebijakan berbasis sumber daya, dan kebijakan tersebut melekat langsung ke fungsi Lambda itu sendiri. Saat membuat aturan yang memanggil fungsi Lambda, Anda tidak lulus peran, sehingga pengguna yang membuat aturan tidak memerlukan izin `iam:PassRole`. Untuk informasi selengkapnya tentang otorisasi fungsi Lambda, lihat [Kelola Izin: Menggunakan Kebijakan Fungsi Lambda](#).

## Menggunakan token untuk menetapkan peran kepada pengguna

Untuk pengguna yang masuk melalui kolam pengguna Amazon Cognito, peran dapat dilewatkan di token ID yang ditetapkan oleh kolam pengguna. Peran muncul dalam klaim berikut di token ID:

- Klaim `cognito:preferred_role` adalah ARN peran.
- Parameter klaim `cognito:roles` adalah string dipisahkan koma yang berisi satu set ARN peran yang diizinkan.

Klaim ditetapkan sebagai berikut:

- Klaim `cognito:preferred_role` diatur untuk peran dari grup dengan nilai yang terbaik (terendah) Precedence. Jika hanya ada satu peran diizinkan, `cognito:preferred_role` diatur ke peran itu. Jika ada beberapa peran dan tidak ada peran tunggal yang memiliki contoh terbaik, klaim ini tidak ditetapkan.
- Klaim `cognito:roles` ditetapkan jika ada setidaknya satu peran.

Saat menggunakan token untuk menetapkan peran, jika ada beberapa peran yang dapat ditetapkan ke pengguna, kolam identitas Amazon Cognito (identitas gabungan) memilih peran sebagai berikut:

- Gunakan [GetCredentialsForIdentityCustomRoleArn](#) parameter jika disetel dan cocok dengan peran dalam `cognito:roles` klaim. Jika parameter ini tidak cocok dengan peran `dicognito:roles`, tolak akses.
- Jika klaim `cognito:preferred_role` ditetapkan, gunakan klaim tersebut.
- Jika `cognito:preferred_role` klaim tidak disetel, `cognito:roles` klaim disetel, dan tidak `CustomRoleArn` ditentukan dalam panggilan ke `GetCredentialsForIdentity`, maka pengaturan Resolusi peran di konsol atau `AmbiguousRoleResolution` bidang (dalam `RoleMappings` parameter [SetIdentityPoolRoles](#) API) digunakan untuk menentukan peran yang akan ditetapkan.

## Menggunakan pemetaan berbasis aturan untuk menetapkan peran kepada pengguna

Aturan memungkinkan Anda memetakan klaim dari token penyedia identitas ke IAM role.

Setiap aturan menentukan klaim token (seperti atribut pengguna di token ID dari kolam pengguna Amazon Cognito), jenis kecocokan, nilai, dan IAM role. Jenis pertandingan dapat `Equals`, `NotEqual`, `StartsWith`, atau `Contains`. Jika pengguna memiliki nilai yang cocok untuk klaim, pengguna dapat mengambil peran itu ketika mendapatkan kredensialnya. Misalnya, Anda dapat membuat aturan yang menetapkan IAM role tertentu untuk pengguna dengan nilai atribut kustom `custom:dept` dari Sales.

### Note

Dalam setelan aturan, atribut khusus memerlukan prefiks `custom:` untuk membedakannya dari atribut standar.

Aturan dievaluasi secara berurutan, dan IAM role untuk aturan pencocokan pertama digunakan, kecuali `CustomRoleArn` ditentukan untuk mengesampingkan urutan tersebut. Untuk informasi selengkapnya tentang atribut pengguna di kolam pengguna Amazon Cognito, lihat [Atribut kumpulan pengguna](#).



Anda dapat menetapkan beberapa aturan untuk penyedia autentikasi di konsol kolam identitas (identitas gabungan). Aturan diterapkan secara berurutan. Anda dapat menyeret aturan untuk mengubah urutannya. Aturan pencocokan pertama diutamakan. Jika jenis kecocokan adalah `NotEqual` dan klaim tidak ada, aturan tidak dievaluasi. Jika tidak ada aturan yang cocok, setelah resolusi Peran diterapkan ke salah satu Gunakan peran default yang diautentikasi atau permintaan Tolak.

Di API dan CLI, Anda dapat menentukan peran yang akan ditetapkan ketika tidak ada aturan yang cocok di `AmbiguousRoleResolution` bidang [RoleMapping](#) tipe, yang ditentukan dalam `RoleMappings` parameter API. [SetIdentityPoolRoles](#)

Anda dapat mengatur pemetaan berbasis aturan untuk OpenID Connect (OIDC) dan penyedia identitas SAMP di atau API AWS CLI dengan bidang jenisnya. `RulesConfiguration` [RoleMapping](#) Anda dapat menentukan bidang ini dalam `RoleMappings` parameter [SetIdentityPoolRolesAPI](#). AWS Management Console Saat ini tidak memungkinkan Anda untuk menambahkan aturan untuk penyedia OIDC atau SAMP.

Misalnya, AWS CLI perintah berikut menambahkan aturan yang menetapkan peran `arn:aws:iam::123456789012:role/Sacramento_team_S3_admin` kepada pengguna di lokasi Sacramento Anda yang diautentikasi oleh OIDC iDP:  
`arn:aws:iam::123456789012:oidc-provider/myOIDCIIDP`

```
aws cognito-identity set-identity-pool-roles --region us-east-1 --cli-input-json
file://role-mapping.json
```

Isi dari **role-mapping.json**:

```
{
  "IdentityPoolId": "us-east-1:12345678-corner-cafe-123456790ab",
  "Roles": {
    "authenticated": "arn:aws:iam::123456789012:role/myS3WriteAccessRole",
    "unauthenticated": "arn:aws:iam::123456789012:role/myS3ReadAccessRole"
  },
  "RoleMappings": {
    "arn:aws:iam::123456789012:oidc-provider/myOIDCIIDP": {
      "Type": "Rules",
      "AmbiguousRoleResolution": "AuthenticatedRole",
      "RulesConfiguration": {
        "Rules": [
          {
            "Claim": "locale",
```

```
        "MatchType": "Equals",
        "Value": "Sacramento",
        "RoleARN": "arn:aws:iam::123456789012:role/
Sacramento_team_S3_admin"
    }
  ]
}
}
```

Untuk setiap kumpulan pengguna atau penyedia otentikasi lain yang Anda konfigurasi untuk kumpulan identitas, Anda dapat membuat hingga 25 aturan. Batas ini tidak dapat disesuaikan. Untuk informasi selengkapnya, lihat [Kuota di Amazon Cognito](#).

## Token mengklaim untuk digunakan dalam pemetaan berbasis aturan

### Amazon Cognito

Sebuah token ID Amazon Cognito direpresentasikan sebagai JSON Web Token (JWT). Token berisi klaim tentang identitas pengguna yang terotentikasi, seperti `name`, `family_name`, dan `phone_number`. Untuk informasi selengkapnya tentang klaim standar, lihat [Spesifikasi OpenID Connect](#). Terlepas dari klaim standar, berikut ini adalah klaim tambahan khusus untuk Amazon Cognito:

- `cognito:groups`
- `cognito:roles`
- `cognito:preferred_role`

### Amazon

Klaim berikut, bersama dengan nilai yang mungkin untuk klaim tersebut, dapat digunakan dengan Login with Amazon:

- `iss`: `www.amazon.com`
- `aud`: Identitas Aplikasi
- `sub`: `sub` dari token Login with Amazon

### Facebook

Klaim berikut, bersama dengan nilai yang mungkin untuk klaim tersebut, dapat digunakan dengan Facebook:

- `iss`: graph.facebook.com
- `aud`: Identitas Aplikasi
- `sub`: sub dari token Facebook

## Google

Token Google berisi klaim standar dari [Spesifikasi OpenID Connect](#). Semua klaim dalam token OpenID tersedia untuk pemetaan berbasis aturan. Lihat situs [OpenID Connect](#) Google untuk mempelajari klaim yang tersedia dari token Google.

## Apel

Token Apple berisi klaim standar dari [Spesifikasi OpenID Connect](#). Lihat [Mengautentikasi Pengguna dengan Masuk melalui Apple](#) dalam dokumentasi Apple untuk belajar lebih lanjut tentang klaim yang tersedia dari token Apple. Token Apple tidak selalu berisi `email`.

## OpenID

Semua klaim dalam token Id Terbuka tersedia untuk pemetaan berbasis aturan. Untuk informasi selengkapnya tentang klaim standar, lihat [Spesifikasi OpenID Connect](#). Lihat dokumentasi penyedia OpenID Anda untuk mempelajari klaim tambahan yang tersedia.

## SAM

Klaim diuraikan dari pernyataan SAML yang diterima. Semua klaim yang tersedia dalam pernyataan SAML dapat digunakan dalam pemetaan berbasis aturan.

## Praktik terbaik untuk kontrol akses berbasis peran

### Important

Jika klaim bahwa Anda memetakan ke peran dapat diubah oleh pengguna akhir, setiap pengguna akhir dapat mengambil peran Anda dan menetapkan kebijakan yang sesuai. Hanya klaim peta yang tidak dapat secara langsung ditetapkan oleh pengguna akhir ke peran dengan izin yang ditinggikan. Di kolam pengguna Amazon Cognito, Anda dapat mengatur izin baca dan tulis per aplikasi untuk setiap atribut pengguna.

### Important

Jika Anda menetapkan peran untuk grup di kolam pengguna Amazon Cognito, peran tersebut akan dilewatkan melalui token ID pengguna. Untuk menggunakan peran ini, Anda juga harus mengatur Pilih peran dari token untuk pemilihan peran terautentikasi untuk kolam identitas. Anda dapat menggunakan pengaturan Resolusi peran di konsol dan `RoleMappings` parameter `SetIdentityPoolRoles` API untuk menentukan perilaku default ketika peran yang benar tidak dapat ditentukan dari token.

## Mendapatkan kredensial

Anda dapat menggunakan Amazon Cognito untuk mengirimkan kredensial hak istimewa terbatas sementara ke aplikasi Anda, sehingga pengguna dapat mengakses sumber daya. AWS Bagian ini menjelaskan cara mendapatkan kredensial dan bagaimana cara mengambil identitas Amazon Cognito dari kolam identitas.

Amazon Cognito mendukung identitas baik yang terautentikasi maupun yang tidak terautentikasi. Pengguna yang tidak diautentikasi tidak meminta agar identitas mereka diverifikasi, sehingga peran ini sesuai untuk pengguna tamu aplikasi Anda atau jika pengguna meminta agar identitas mereka diverifikasi dan itu tidak dipermasalahkan. Pengguna yang diautentikasi masuk ke aplikasi Anda melalui penyedia identitas pihak ketiga, atau kolam pengguna, yang memverifikasi identitas mereka. Pastikan Anda menjangkau izin sumber daya dengan tepat sehingga Anda tidak memberikan akses kepada mereka dari pengguna yang tidak terautentikasi.

Identitas Amazon Cognito identitas bukan kredensial. Mereka ditukar dengan kredensial menggunakan dukungan federasi identitas web di (). AWS Security Token Service AWS STS Cara yang disarankan untuk mendapatkan kredensial AWS untuk pengguna aplikasi Anda adalah dengan menggunakan `AWS.CognitoIdentityCredentials`. Identitas dalam objek kredensial kemudian ditukar dengan kredensial menggunakan. AWS STS

### Note

Jika Anda membuat kumpulan identitas sebelum Februari 2015, Anda harus mengasosiasikan kembali peran Anda dengan kumpulan identitas Anda untuk menggunakan `AWS.CognitoIdentityCredentials` konstruktor tanpa peran sebagai parameter. Untuk melakukannya, buka [konsol Amazon Cognito](#), pilih Kelola kumpulan identitas, pilih kumpulan

identitas Anda, pilih Edit Kumpulan identitas, tentukan peran yang diautentikasi dan tidak diautentikasi, dan simpan perubahan.

Penyedia kredensial identitas web adalah bagian dari rantai penyedia kredensi default di SDK. AWS Untuk menyetel token kumpulan identitas Anda dalam config file lokal untuk AWS SDK atau file AWS CLI, tambahkan entri `web_identity_token_file` profil. Lihat [Mengasumsikan penyedia kredensi peran](#) dalam Panduan Referensi AWS SDK dan Alat.

Untuk mempelajari lebih lanjut tentang cara mengisi kredensial identitas web di SDK Anda, lihat panduan pengembang SDK. Untuk hasil terbaik, mulailah proyek Anda dengan integrasi kumpulan identitas yang ada di dalamnya AWS Amplify.

AWS Sumber daya SDK untuk mendapatkan dan menyetel kredensial dengan kumpulan identitas

- [Federasi Kolam Identitas](#) (Android) di Amplify Dev Center
- [Federasi Kolam Identitas](#) (iOS) di Amplify Dev Center
- [Menggunakan Identitas Amazon Cognito untuk mengautentikasi pengguna di Panduan Pengembang](#) AWS SDK for JavaScript
- [Penyedia kredensi Amazon Cognito](#) di Panduan Pengembang AWS SDK for .NET
- [Tentukan Kredensial Secara Terprogram di Panduan Pengembang](#) AWS SDK for Go
- [Menyediakan kredensial sementara dalam kode di](#) Panduan Pengembang AWS SDK for Java 2.x
- [assumeRoleWithWebIdentityCredentialProvider](#) penyedia di Panduan AWS SDK for PHP Pengembang
- [Asumsikan Peran Dengan Penyedia Identitas Web](#) dalam AWS SDK for Python (Boto3) dokumentasi
- [Menentukan kredensial dan wilayah default Anda](#) di Panduan Pengembang AWS SDK for Rust

Bagian berikut memberikan contoh kode di beberapa AWS SDK lama.

## Android

Anda dapat menggunakan Amazon Cognito untuk mengirimkan kredensial hak istimewa terbatas sementara ke aplikasi Anda, sehingga pengguna dapat mengakses sumber daya. AWS Amazon Cognito mendukung identitas baik yang terautentikasi maupun yang tidak terautentikasi. Untuk memberikan AWS kredensial ke aplikasi Anda, ikuti langkah-langkah di bawah ini.

Untuk menggunakan kumpulan identitas Amazon Cognito di aplikasi Android, siapkan. AWS Amplify Untuk informasi selengkapnya, lihat [Otentikasi](#) di Amplify Dev Center.

### Mengambil identitas Amazon Cognito

Jika Anda mengizinkan pengguna yang tidak diautentikasi, Anda dapat segera mengambil pengidentifikasi Amazon Cognito (ID identitas) unik untuk pengguna akhir Anda. Jika Anda mengautentikasi pengguna, Anda dapat mengambil ID identitas setelah menyetel token masuk di penyedia kredensial:

```
String identityId = credentialsProvider.getIdentityId();
Log.d("LogTag", "my ID is " + identityId);
```

#### Note

Jangan panggil `getIdentityId()`, `refresh()`, atau `getCredentials()` di thread utama aplikasi Anda. Mulai Android 3.0 (API Level 11), aplikasi Anda akan secara otomatis gagal dan menampilkan I/O jaringan [NetworkOnMainThreadException](#) jika Anda menjalankan I/O jaringan di utas aplikasi utama. Anda harus memindahkan kode Anda ke thread latar belakang menggunakan `AsyncTask`. Untuk informasi lebih lanjut, lihat [dokumentasi Android](#). Anda juga dapat memanggil `getCachedIdentityId()` untuk menarik ID, tetapi hanya jika hal tersebut sudah di-cache secara lokal. Jika tidak, metode ini akan memberikan nilai null.

## iOS - Objective-C

Anda dapat menggunakan Amazon Cognito untuk mengirimkan kredensial hak istimewa terbatas sementara ke aplikasi Anda, sehingga pengguna dapat mengakses sumber daya. AWS Kolam identitas Amazon Cognito mendukung identitas terautentikasi dan tidak terautentikasi. Untuk memberikan AWS kredensi ke aplikasi Anda, selesaikan langkah-langkah berikut.

Untuk menggunakan kumpulan identitas Amazon Cognito di aplikasi iOS, siapkan. AWS Amplify Untuk informasi selengkapnya, lihat Otentikasi [Swift dan Otentikasi Flutter di](#) Amplify Dev Center.

### Mengambil identitas Amazon Cognito

Anda dapat segera menarik pengidentifikasi Amazon Cognito unik (ID identitas) untuk pengguna akhir Anda jika Anda mengizinkan pengguna yang tidak diautentikasi atau setelah Anda menyetel token login di penyedia kredensial jika Anda mengautentikasi pengguna:

```
// Retrieve your Amazon Cognito ID
[[credentialsProvider getIdentityId] continueWithBlock:^id(AWSTask *task) {
    if (task.error) {
        NSLog(@"Error: %@", task.error);
    }
    else {
        // the task result will contain the identity id
        NSString *cognitoId = task.result;
    }
    return nil;
}];
```

### Note

`getIdentityId` adalah panggilan asinkron. Jika ID identitas telah diatur pada penyedia layanan Anda, Anda dapat menghubungi `credentialsProvider.identityId` untuk menarik identitas itu, yang di-cache secara di lokasi tersebut. Namun, jika ID identitas tidak diatur pada penyedia Anda, memanggil `credentialsProvider.identityId` akan menampilkan `nil`. Untuk informasi selengkapnya, lihat referensi [Amplify iOS SDK](#).

## iOS - Swift

Anda dapat menggunakan Amazon Cognito untuk mengirimkan kredensial hak istimewa terbatas sementara ke aplikasi Anda sehingga pengguna dapat mengakses sumber daya. AWS Amazon Cognito mendukung identitas baik yang terautentikasi maupun yang tidak terautentikasi. Untuk memberikan AWS kredensi ke aplikasi Anda, ikuti langkah-langkah di bawah ini.

Untuk menggunakan kumpulan identitas Amazon Cognito di aplikasi iOS, siapkan. AWS Amplify Untuk informasi selengkapnya, lihat [Otentikasi Swift](#) di Amplify Dev Center.

### Mengambil identitas Amazon Cognito

Anda dapat segera menarik pengidentifikasi Amazon Cognito unik (ID identitas) untuk pengguna akhir Anda jika Anda mengizinkan pengguna yang tidak diautentikasi atau setelah Anda menyetel token login di penyedia kredensial jika Anda mengautentikasi pengguna:

```
// Retrieve your Amazon Cognito ID
credentialsProvider.getIdentityId().continueWith(block: { (task) -> AnyObject? in
    if (task.error != nil) {
```

```
        print("Error: " + task.error!.localizedDescription)
    }
    else {
        // the task result will contain the identity id
        let cognitoId = task.result!
        print("Cognito id: \(cognitoId)")
    }
    return task;
})
```

### Note

`getIdentityId` adalah panggilan asinkron. Jika ID identitas telah diatur pada penyedia layanan Anda, Anda dapat menghubungi `credentialsProvider.identityId` untuk menarik identitas itu, yang di-cache secara di lokasi tersebut. Namun, jika ID identitas tidak diatur pada penyedia Anda, memanggil `credentialsProvider.identityId` akan menampilkan `nil`. Untuk informasi selengkapnya, lihat referensi [Amplify iOS SDK](#).

## JavaScript

Jika Anda belum membuatnya, buat kolam identitas di [Konsol Amazon Cognito](#) sebelum menggunakan `AWS.CognitoIdentityCredentials`.

Setelah mengkonfigurasi kolam identitas dengan penyedia identitas, Anda dapat menggunakan `AWS.CognitoIdentityCredentials` untuk mengautentikasi pengguna. Untuk mengkonfigurasi kredensial aplikasi Anda agar dapat menggunakan `AWS.CognitoIdentityCredentials`, atur properti `credentials` baik `AWS.Config` atau konfigurasi per layanan. Contoh berikut menggunakan `AWS.Config`:

```
// Set the region where your identity pool exists (us-east-1, eu-west-1)
AWS.config.region = 'us-east-1';

// Configure the credentials provider to use your identity pool
AWS.config.credentials = new AWS.CognitoIdentityCredentials({
  IdentityPoolId: 'IDENTITY_POOL_ID',
  Logins: { // optional tokens, used for authenticated login
    'graph.facebook.com': 'FBTOKEN',
    'www.amazon.com': 'AMAZONTOKEN',
    'accounts.google.com': 'GOOGLETOKEN',
    'appleid.apple.com': 'APPLETOKEN'
  }
});
```



```
    }
  });

  // Make the call to obtain credentials
  AWS.config.credentials.get(function(){

    // Credentials will be available when this function is called.
    var accessKeyId = AWS.config.credentials.accessKeyId;
    var secretAccessKey = AWS.config.credentials.secretAccessKey;
    var sessionToken = AWS.config.credentials.sessionToken;

  });
```

Properti opsional Logins adalah peta nama penyedia identitas untuk token identitas bagi penyedia tersebut. Bagaimana Anda bisa mendapatkan token dari penyedia identitas Anda tergantung pada penyedia yang Anda gunakan. Misalnya, jika Facebook adalah salah satu penyedia identitas Anda, Anda dapat menggunakan fungsi `FB.login` dari [SDK Facebook](#) untuk mendapatkan token penyedia identitas:

```
FB.login(function (response) {
  if (response.authResponse) { // logged in
    AWS.config.credentials = new AWS.CognitoIdentityCredentials({
      IdentityPoolId: 'us-east-1:1699ebc0-7900-4099-b910-2df94f52a030',
      Logins: {
        'graph.facebook.com': response.authResponse.accessToken
      }
    });

    console.log('You are now logged in.');
```

```
  } else {
    console.log('There was a problem logging you in.');
```

```
  }
});
```

## Mengambil identitas Amazon Cognito

Anda dapat segera menarik pengidentifikasi Amazon Cognito unik (ID identitas) untuk pengguna akhir Anda jika Anda mengizinkan pengguna yang tidak diautentikasi atau setelah Anda menyetel token login di penyedia kredensial jika Anda mengautentikasi pengguna:

```
var identityId = AWS.config.credentials.identityId;
```

## Unity

Anda dapat menggunakan Amazon Cognito untuk mengirimkan kredensial hak istimewa terbatas sementara ke aplikasi Anda, sehingga pengguna dapat mengakses sumber daya. AWS Amazon Cognito mendukung identitas baik yang terautentikasi maupun yang tidak terautentikasi. Untuk memberikan AWS kredensi ke aplikasi Anda, ikuti langkah-langkah di bawah ini.

[AWS SDK for Unity](#) sekarang menjadi bagian dari [AWS SDK for .NET](#) Untuk memulai dengan Amazon Cognito di AWS SDK for .NET, lihat Penyedia [kredensi Amazon Cognito](#) di Panduan Pengembang. AWS SDK for .NET Atau lihat [Amplify Dev Center](#) untuk opsi untuk membuat aplikasi. AWS Amplify

### Mengambil identitas Amazon Cognito

Anda dapat segera menarik pengidentifikasi Amazon Cognito unik (ID identitas) untuk pengguna akhir Anda jika Anda mengizinkan pengguna yang tidak diautentikasi atau setelah Anda menyetel token login di penyedia kredensial jika Anda mengautentikasi pengguna:

```
credentials.GetIdentityIdAsync(delegate(AmazonCognitoIdentityResult<string> result) {
    if (result.Exception != null) {
        //Exception!
    }
    string identityId = result.Response;
});
```

## Xamarin

Anda dapat menggunakan Amazon Cognito untuk mengirimkan kredensial hak istimewa terbatas sementara ke aplikasi Anda sehingga pengguna dapat mengakses sumber daya. AWS Amazon Cognito mendukung identitas baik yang terautentikasi maupun yang tidak terautentikasi. Untuk memberikan AWS kredensi ke aplikasi Anda, ikuti langkah-langkah di bawah ini.

[AWS SDK untuk Xamarin](#) sekarang menjadi bagian dari [AWS SDK for .NET](#) Untuk memulai dengan Amazon Cognito di AWS SDK for .NET, lihat Penyedia [kredensi Amazon Cognito](#) di Panduan Pengembang. AWS SDK for .NET Atau lihat [Amplify Dev Center](#) untuk opsi untuk membuat aplikasi. AWS Amplify

### Note

Catatan: Jika Anda membuat kumpulan identitas sebelum Februari 2015, Anda harus mengasosiasikan kembali peran Anda dengan kumpulan identitas Anda untuk menggunakan

konstruktor ini tanpa peran sebagai parameter. Untuk melakukannya, buka [konsol Amazon Cognito](#), pilih Kelola kumpulan identitas, pilih kumpulan identitas Anda, pilih Edit Kumpulan identitas, tentukan peran yang diautentikasi dan tidak diautentikasi, dan simpan perubahan.

## Mengambil identitas Amazon Cognito

Anda dapat segera menarik pengidentifikasi Amazon Cognito unik (ID identitas) untuk pengguna akhir Anda jika Anda mengizinkan pengguna yang tidak diautentikasi atau setelah Anda menyetel token login di penyedia kredensial jika Anda mengautentikasi pengguna:

```
var identityId = await credentials.GetIdentityIdAsync();
```

## Mengakses layanan AWS

Setelah mengonfigurasi penyedia kredensi Amazon Cognito dan mengambil AWS kredensial, Anda dapat membuat klien. Layanan AWS

AWS Sumber daya SDK untuk membuat klien

- [AWS Konfigurasi klien](#) di Panduan AWS SDK for C++ Pengembang
- [Menggunakan AWS SDK for Go V2 dengan Layanan AWS](#) di Panduan AWS SDK for Go Pengembang
- [Mengkonfigurasi klien HTTP](#) di Panduan AWS SDK for Java 2.x Pengembang
- [Membuat dan memanggil objek layanan](#) di Panduan AWS SDK for JavaScript Pengembang
- [Membuat klien](#) dalam AWS SDK for Python (Boto3) dokumentasi
- [Membuat klien layanan](#) di Panduan AWS SDK for Rust Pengembang
- [Menggunakan klien](#) di Panduan AWS SDK for Swift Pengembang

Cuplikan berikut menginisialisasi klien Amazon DynamoDB:

## Android

Untuk menggunakan kumpulan identitas Amazon Cognito di aplikasi Android, siapkan. AWS Amplify Untuk informasi selengkapnya, lihat [Otentikasi](#) di Amplify Dev Center.

```
// Create a service client with the provider
AmazonDynamoDB client = new AmazonDynamoDBClient(credentialsProvider);
```

Penyedia kredensi berkomunikasi dengan Amazon Cognito, mengambil pengenal unik untuk pengguna yang diautentikasi dan tidak diautentikasi serta kredensial hak istimewa sementara dan terbatas untuk SDK Seluler. AWS AWS Kredensial yang ditarik berlaku selama satu jam, dan penyedia menyegarkannya ketika kedaluwarsa.

## iOS - Objective-C

Untuk menggunakan kumpulan identitas Amazon Cognito di aplikasi iOS, siapkan. AWS Amplify Untuk informasi selengkapnya, lihat Otentikasi [Swift](#) dan Otentikasi [Flutter](#) di Amplify Dev Center.

```
// create a configuration that uses the provider
AWSServiceConfiguration *configuration = [AWSServiceConfiguration
    configurationWithRegion:AWSRegionUSEast1 provider:credentialsProvider];
// get a client with the default service configuration
AWSDynamoDB *dynamoDB = [AWSDynamoDB defaultDynamoDB];
```

Penyedia kredensi berkomunikasi dengan Amazon Cognito, mengambil pengenal unik untuk pengguna yang diautentikasi dan tidak diautentikasi serta kredensial hak istimewa sementara dan terbatas untuk SDK Seluler. AWS AWS Kredensial yang ditarik berlaku selama satu jam, dan penyedia menyegarkannya ketika kedaluwarsa.

## iOS - Swift

Untuk menggunakan kumpulan identitas Amazon Cognito di aplikasi iOS, siapkan. AWS Amplify Untuk informasi selengkapnya, lihat [Otentikasi Swift](#) di Amplify Dev Center.

```
// get a client with the default service configuration
let dynamoDB = AWSDynamoDB.default()

// get a client with a custom configuration
AWSDynamoDB.register(with: configuration!, forKey: "USWest2DynamoDB");
let dynamoDBCustom = AWSDynamoDB(forKey: "USWest2DynamoDB")
```

Penyedia kredensi berkomunikasi dengan Amazon Cognito, mengambil pengenal unik untuk pengguna yang diautentikasi dan tidak diautentikasi serta kredensial hak istimewa sementara dan

terbatas untuk SDK Seluler. AWS AWS Kredensial yang ditarik berlaku selama satu jam, dan penyedia menyegarkannya ketika kedaluwarsa.

## JavaScript

```
// Create a service client with the provider
var dynamodb = new AWS.DynamoDB({region: 'us-west-2'});
```

Penyedia kredensi berkomunikasi dengan Amazon Cognito, mengambil pengenal unik untuk pengguna yang diautentikasi dan tidak diautentikasi serta kredensial hak istimewa terbatas sementara untuk SDK Seluler. AWS AWS Kredensial yang ditarik berlaku selama satu jam, dan penyedia menyegarkannya ketika kedaluwarsa.

## Unity

[AWS SDK for Unity](#) sekarang menjadi bagian dari [AWS SDK for .NET](#) Untuk memulai dengan Amazon Cognito di AWS SDK for .NET, lihat Penyedia [kredensial Amazon Cognito](#) di Panduan Pengembang. AWS SDK for .NET Atau lihat [Amplify Dev Center](#) untuk opsi untuk membuat aplikasi. AWS Amplify

```
// create a service client that uses credentials provided by Cognito
AmazonDynamoDBClient client = new AmazonDynamoDBClient(credentials, REGION);
```

Penyedia kredensi berkomunikasi dengan Amazon Cognito, mengambil pengenal unik untuk pengguna yang diautentikasi dan tidak diautentikasi serta kredensial hak istimewa terbatas sementara untuk SDK Seluler. AWS AWS Kredensial yang ditarik berlaku selama satu jam, dan penyedia menyegarkannya ketika kedaluwarsa.

## Xamarin

[AWS SDK untuk Xamarin](#) sekarang menjadi bagian dari [AWS SDK for .NET](#) Untuk memulai dengan Amazon Cognito di AWS SDK for .NET, lihat Penyedia [kredensial Amazon Cognito](#) di Panduan Pengembang. AWS SDK for .NET Atau lihat [Amplify Dev Center](#) untuk opsi pembuatan aplikasi. AWS Amplify

```
// create a service client that uses credentials provided by Cognito
var client = new AmazonDynamoDBClient(credentials, REGION)
```

Penyedia kredensi berkomunikasi dengan Amazon Cognito, mengambil pengenal unik untuk pengguna yang diautentikasi dan tidak diautentikasi serta kredensial hak istimewa terbatas sementara untuk SDK Seluler. AWS Kredsial yang ditarik berlaku selama satu jam, dan penyedia menyegarkannya ketika kedaluwarsa.

## Identitas mengumpulkan penyedia identitas eksternal

Menggunakan `logins` properti, Anda dapat mengatur kredensial yang diterima dari penyedia identitas (iDP). Selanjutnya, Anda dapat mengaitkan kumpulan identitas dengan beberapa IdPs. Misalnya, Anda dapat mengatur token Facebook dan Google di `logins` properti untuk mengaitkan identitas Amazon Cognito yang unik dengan kedua login iDP. Pengguna dapat mengautentikasi dengan salah satu akun, tetapi Amazon Cognito mengembalikan pengenal pengguna yang sama.

Petunjuk berikut memandu Anda melalui otentikasi dengan dukungan kumpulan IdPs identitas Amazon Cognito.

### Topik

- [Menyiapkan Facebook sebagai kumpulan identitas IDP](#)
- [Menyiapkan Login with Amazon sebagai kumpulan identitas IDP](#)
- [Menyiapkan Google sebagai kumpulan identitas iDP](#)
- [Menyiapkan Masuk dengan Apple sebagai kumpulan identitas iDP](#)
- [Menyiapkan penyedia OIDC sebagai idP kumpulan identitas](#)
- [Menyiapkan penyedia SAMP sebagai idP kumpulan identitas](#)

## Menyiapkan Facebook sebagai kumpulan identitas IDP

Kolam identitas Amazon Cognito mengintegrasikan dengan Facebook untuk menyediakan autentikasi federasi untuk pengguna aplikasi seluler Anda. Bagian ini menjelaskan cara mendaftar dan mengatur aplikasi Anda dengan Facebook sebagai IDP.

### Mengatur Facebook

Daftarkan aplikasi Anda dengan Facebook sebelum Anda mengautentikasi pengguna Facebook dan berinteraksi dengan API Facebook.

[Portal Pengembang Facebook](#) membantu Anda mengatur aplikasi Anda. Lakukan prosedur ini sebelum Anda mengintegrasikan Facebook di kumpulan identitas Amazon Cognito Anda:

## Menyiapkan Facebook

1. Pada [Portal Developer Facebook](#), masuk dengan kredensial Facebook Anda.
2. Dari menu Aplikasi, pilih Tambah Aplikasi Baru.
3. Pilih platform dan selesaikan proses mulai cepat.

### Android

Untuk informasi selengkapnya tentang cara mengintegrasikan aplikasi Android dengan Login Facebook, lihat [Panduan Memulai Facebook](#).

### iOS - Objective-C

Untuk informasi selengkapnya tentang cara mengintegrasikan aplikasi iOS Objective-C dengan Login Facebook, lihat Panduan [Memulai Facebook](#).

### iOS - Swift

Untuk informasi selengkapnya tentang cara mengintegrasikan aplikasi iOS Swift dengan Login Facebook, lihat [Panduan Memulai Facebook](#).

### JavaScript

Untuk informasi selengkapnya tentang cara mengintegrasikan aplikasi JavaScript web dengan Login Facebook, lihat [Panduan Memulai Facebook](#).

### Unity

Untuk informasi selengkapnya tentang cara mengintegrasikan aplikasi Unity dengan Login Facebook, lihat [Panduan Memulai Facebook](#).

### Xamarin

Untuk menambahkan otentikasi Facebook, pertama-tama ikuti alur yang sesuai di bawah ini untuk mengintegrasikan SDK Facebook ke dalam aplikasi Anda. Kolam identitas Amazon Cognito menggunakan token akses Facebook untuk menghasilkan pengidentifikasi pengguna unik yang dikaitkan dengan identitas Amazon Cognito.

- [SDK iOS Facebook oleh Xamarin](#)
- [SDK Android Facebook oleh Xamarin](#)

## Mengonfigurasi penyedia identitas di konsol kumpulan identitas Amazon Cognito

Gunakan prosedur berikut untuk mengonfigurasi penyedia identitas Anda.

Untuk menambahkan penyedia identitas Facebook (iDP)

1. Pilih kumpulan Identitas dari konsol [Amazon Cognito](#). Pilih kumpulan identitas.
2. Pilih tab Akses pengguna.
3. Pilih Tambahkan penyedia identitas.
4. Pilih Facebook.
5. Masukkan ID Aplikasi proyek OAuth yang Anda buat di [Meta](#) for Developers. Untuk informasi selengkapnya, lihat [Login Facebook](#) di Meta for Developers Docs.
6. Untuk menyetel peran yang diminta Amazon Cognito saat mengeluarkan kredensial kepada pengguna yang telah mengautentikasi dengan penyedia ini, konfigurasi setelan Peran.
  - Anda dapat menetapkan pengguna dari IDP tersebut peran Default yang Anda atur saat mengonfigurasi peran Terautentikasi, atau Anda dapat Memilih peran dengan aturan.
    - i. Jika Anda memilih Pilih peran dengan aturan, masukkan Klaim sumber dari autentikasi pengguna Anda, Operator yang ingin Anda bandingkan dengan klaim, Nilai yang akan menyebabkan kecocokan dengan pilihan peran ini, dan Peran yang ingin Anda tetapkan saat penetapan Peran cocok. Pilih Tambahkan yang lain untuk membuat aturan tambahan berdasarkan kondisi yang berbeda.
    - ii. Pilih Resolusi Peran. Jika klaim pengguna tidak sesuai dengan aturan, Anda dapat menolak kredensial atau mengeluarkan kredensial untuk peran yang Diautentikasi.
7. Untuk mengubah tag utama yang ditetapkan Amazon Cognito saat mengeluarkan kredensial kepada pengguna yang telah diautentikasi dengan penyedia ini, konfigurasi Atribut untuk kontrol akses.
  - a. Untuk tidak menerapkan tag utama, pilih Tidak aktif.
  - b. Untuk menerapkan tag utama berdasarkan sub dan aud klaim, pilih Gunakan pemetaan default.
  - c. Untuk membuat skema atribut kustom Anda sendiri ke tag utama, pilih Gunakan pemetaan khusus. Kemudian masukkan kunci Tag yang ingin Anda sumber dari setiap Klaim yang ingin Anda wakili dalam tag.
8. Pilih Simpan perubahan.



## Menggunakan Facebook

### Android

Untuk menambahkan otentikasi Facebook, pertama-tama ikuti [panduan Facebook](#) dan integrasikan SDK Facebook ke dalam aplikasi Anda. Kemudian tambahkan [tombol Login dengan Facebook](#) ke antarmuka pengguna Android Anda. SDK Facebook menggunakan objek sesi untuk melacak keadaannya. Amazon Cognito menggunakan token akses dari objek sesi ini untuk mengautentikasi pengguna, menghasilkan pengenal unik, dan, jika diperlukan, memberikan pengguna akses ke sumber daya lain. AWS

Setelah Anda mengautentikasi pengguna Anda dengan Facebook SDK, tambahkan token sesi ke penyedia kredensi Amazon Cognito.

SDK Facebook 4.0 atau lebih baru:

```
Map<String, String> logins = new HashMap<String, String>();
logins.put("graph.facebook.com", AccessToken.getCurrentAccessToken().getToken());
credentialsProvider.setLogins(logins);
```

SDK Facebook sebelum 4.0:

```
Map<String, String> logins = new HashMap<String, String>();
logins.put("graph.facebook.com", Session.getActiveSession().getAccessToken());
credentialsProvider.setLogins(logins);
```

Proses login Facebook menginisialisasi sesi tunggal di SDK-nya. Objek sesi Facebook berisi token OAuth yang digunakan Amazon Cognito untuk AWS menghasilkan kredensial bagi pengguna akhir Anda yang diautentikasi. Amazon Cognito juga menggunakan token untuk memeriksa database pengguna Anda untuk keberadaan pengguna yang cocok dengan identitas Facebook khusus ini. Jika pengguna sudah ada, API mengembalikan pengidentifikasi yang ada. Jika tidak, API mengembalikan pengenal baru. SDK klien secara otomatis menyimpan pengenal di perangkat lokal.

#### Note

Setelah Anda mengatur peta login, lakukan panggilan ke `refresh` atau `get` untuk mengambil kredensialnya. AWS

## iOS - Objective-C

Untuk menambahkan otentikasi Facebook, pertama-tama ikuti [panduan Facebook](#) dan integrasikan SDK Facebook ke dalam aplikasi Anda. Kemudian tambahkan [Tombol Masuk dengan Facebook](#) ke antarmuka pengguna Anda. SDK Facebook menggunakan objek sesi untuk melacak keadaannya. Amazon Cognito menggunakan token akses dari objek sesi ini untuk mengautentikasi pengguna dan mengikat mereka ke kolam identitas Amazon Cognito unik (identitas federasi).

Untuk menyediakan token akses Facebook ke Amazon Cognito, terapkan protokol.

### [AWSIdentityProviderManager](#)

Saat Anda menerapkan `logins` metode ini, kembalikan kamus yang berisi `AWSIdentityProviderFacebook`. Kamus ini bertindak sebagai kunci, dan token akses saat ini dari pengguna Facebook yang diautentikasi bertindak sebagai nilai, seperti yang ditunjukkan pada contoh kode berikut.

```
- (AWSTask<NSDictionary<NSString *, NSString *> *> *)logins {
    FBSDKAccessToken* fbToken = [FBSDKAccessToken currentAccessToken];
    if(fbToken){
        NSString *token = fbToken.tokenString;
        return [AWSTask taskWithResult: @{ AWSIdentityProviderFacebook : token }];
    }else{
        return [AWSTask taskWithError:[NSError errorWithDomain:@"Facebook Login"
                                                    code:-1
                                                    userInfo:@{@"error":@"No current
Facebook access token"}]];
    }
}
```

Saat Anda memberi contoh `AWSCognitoCredentialsProvider`, teruskan kelas yang menerapkan `AWSIdentityProviderManager` sebagai nilai dari `identityProviderManager` dalam konstruktor. Untuk informasi lebih lanjut, buka halaman [AWSCognitoCredentialsProvider](#) referensi dan pilih `initWithRegionJenis:identityPoolId:identityProviderManager`.

## iOS - Swift

Untuk menambahkan otentikasi Facebook, pertama-tama ikuti [panduan Facebook](#) dan integrasikan SDK Facebook ke dalam aplikasi Anda. Kemudian tambahkan [Tombol Masuk dengan Facebook](#) ke antarmuka pengguna Anda. SDK Facebook menggunakan objek sesi untuk melacak keadaannya.

Amazon Cognito menggunakan token akses dari objek sesi ini untuk mengautentikasi pengguna dan mengikat mereka ke kolam identitas Amazon Cognito unik (identitas federasi).

Untuk menyediakan token akses Facebook ke Amazon Cognito, terapkan protokol.

### [AWSIdentityProviderManager](#)

Saat Anda menerapkan logins metode ini, kembalikan kamus yang berisi `AWSIdentityProviderFacebook`. Kamus ini bertindak sebagai kunci, dan token akses saat ini dari pengguna Facebook yang diautentikasi bertindak sebagai nilai, seperti yang ditunjukkan pada contoh kode berikut.

```
class FacebookProvider: NSObject, AWSIdentityProviderManager {
    func logins() -> AWSTask<NSDictionary> {
        if let token = AccessToken.current?.authenticationToken {
            return AWSTask(result: [AWSIdentityProviderFacebook:token])
        }
        return AWSTask(error: NSError(domain: "Facebook Login", code: -1 , userInfo:
["Facebook" : "No current Facebook access token"]))
    }
}
```

Saat Anda memberi contoh `AWSCognitoCredentialsProvider`, teruskan kelas yang menerapkan `AWSIdentityProviderManager` sebagai nilai dari `identityProviderManager` dalam konstruktor. Untuk informasi lebih lanjut, buka halaman [AWSCognitoCredentialsProvider](#) referensi dan pilih `initWithRegionJenis:identityPoolId:identityProviderManager`.

### JavaScript

Untuk menambahkan otentikasi Facebook, ikuti [Login Facebook untuk Web](#) dan tambahkan tombol Login dengan Facebook di situs web Anda. SDK Facebook menggunakan objek sesi untuk melacak keadaannya. Amazon Cognito menggunakan token akses dari objek sesi ini untuk mengautentikasi pengguna, menghasilkan pengenal unik, dan, jika diperlukan, memberikan pengguna akses ke sumber daya lain. AWS

Setelah Anda mengautentikasi pengguna Anda dengan Facebook SDK, tambahkan token sesi ke penyedia kredensi Amazon Cognito.

```
FB.login(function (response) {

    // Check if the user logged in successfully.
```

```
if (response.authResponse) {

    console.log('You are now logged in.');
```

```
    // Add the Facebook access token to the Amazon Cognito credentials login map.
    AWS.config.credentials = new AWS.CognitoIdentityCredentials({
        IdentityPoolId: 'IDENTITY_POOL_ID',
        Logins: {
            'graph.facebook.com': response.authResponse.accessToken
        }
    });

    // Obtain AWS credentials
    AWS.config.credentials.get(function(){
        // Access AWS resources here.
    });

} else {
    console.log('There was a problem logging you in.');
```

```
}

});
```

SDK Facebook memperoleh token OAuth yang digunakan Amazon Cognito untuk menghasilkan AWS kredensial bagi pengguna akhir Anda yang diautentikasi. Amazon Cognito juga menggunakan token untuk memeriksa pada basis data pengguna Anda untuk keberadaan pengguna yang cocok dengan identitas Facebook tertentu ini. Jika pengguna sudah ada, API mengembalikan pengidentifikasi yang ada. Jika tidak sebuah pengidentifikasi baru dikembalikan. Pengidentifikasi secara otomatis di-cache oleh SDK klien di perangkat lokal.

#### Note

Setelah Anda mengatur peta login, lakukan panggilan ke `refresh` atau `get` untuk mendapatkan kredensialnya. [Untuk contoh kode, lihat “Gunakan Kasus 17, Mengintegrasikan Kumpulan Pengguna dengan Identitas Cognito,” di JavaScript file README.](#)

## Unity

Untuk menambahkan otentikasi Facebook, pertama-tama ikuti [panduan Facebook](#) dan integrasikan SDK Facebook ke dalam aplikasi Anda. Amazon Cognito menggunakan token akses Facebook

dari FB objek untuk menghasilkan pengenalan pengguna unik yang terkait dengan identitas Amazon Cognito.

Setelah Anda mengautentikasi pengguna Anda dengan Facebook SDK, tambahkan token sesi ke penyedia kredensi Amazon Cognito:

```
void Start()
{
    FB.Init(delegate() {
        if (FB.IsLoggedIn) { //User already logged in from a previous session
            AddFacebookTokenToCognito();
        } else {
            FB.Login ("email", FacebookLoginCallback);
        }
    });
}

void FacebookLoginCallback(FBResult result)
{
    if (FB.IsLoggedIn)
    {
        AddFacebookTokenToCognito();
    }
    else
    {
        Debug.Log("FB Login error");
    }
}

void AddFacebookTokenToCognito()
{
    credentials.AddLogin ("graph.facebook.com",
        AccessToken.CurrentAccessToken.TokenString);
}
```

Sebelum Anda menggunakan `FB.AccessToken`, hubungi `FB.Login()` dan pastikan `FB.IsLoggedIn` itu benar.

Xamarin

Xamarin untuk Android:

```
public void InitializeFacebook() {
```

```
FacebookSdk.SdkInitialize(this.ApplicationContext);
callbackManager = CallbackManagerFactory.Create();
LoginManager.Instance.RegisterCallback(callbackManager, new FacebookCallback <<
LoginResult >> () {
    HandleSuccess = loginResult = >> {
        var accessToken = loginResult.AccessToken;
        credentials.AddLogin("graph.facebook.com", accessToken.Token);
        //open new activity
    },
    HandleCancel = () = >> {
        //throw error message
    },
    HandleError = loginError = >> {
        //throw error message
    }
});
LoginManager.Instance.LogInWithReadPermissions(this, new List << string >> {
    "public_profile"
});
}
```

## Xamarin untuk iOS:

```
public void InitializeFacebook() {
    LoginManager login = new LoginManager();
    login.LogInWithReadPermissions(readPermissions.ToArray(),
    delegate(LoginManagerLoginResult result, NSError error) {
        if (error != null) {
            //throw error message
        } else if (result.IsCancelled) {
            //throw error message
        } else {
            var accessToken = loginResult.AccessToken;
            credentials.AddLogin("graph.facebook.com", accessToken.Token);
            //open new view controller
        }
    });
}
```

## Menyiapkan Login with Amazon sebagai kumpulan identitas IDP

Amazon Cognito terintegrasi dengan Login with Amazon untuk memberikan otentikasi federasi bagi pengguna aplikasi seluler dan web Anda. Bagian ini menjelaskan cara mendaftar dan menyiapkan aplikasi Anda dengan Login with Amazon sebagai penyedia identitas (iDP).

Siapkan Login with Amazon untuk bekerja dengan Amazon Cognito di Portal [Pengembang](#). Untuk informasi selengkapnya, lihat [Menyiapkan Login with Amazon](#) di FAQ Login with Amazon.

### Note

Untuk mengintegrasikan Login with Amazon ke dalam aplikasi Xamarin, ikuti Panduan Memulai [Xamarin](#).

### Note

Anda tidak dapat mengintegrasikan Login with Amazon secara native di platform Unity. Sebagai gantinya, gunakan tampilan web dan buka alur masuk browser.

## Menyiapkan Login with Amazon

### Menerapkan Login with Amazon

Di [portal pengembang Amazon](#), Anda dapat menyiapkan aplikasi OAuth untuk diintegrasikan dengan kumpulan identitas Anda, menemukan dokumentasi Login with Amazon, dan mengunduh SDK. Pilih Konsol pengembang, lalu Login with Amazon di portal pengembang. Anda dapat membuat profil keamanan untuk aplikasi Anda dan kemudian membangun Login with Amazon mekanisme autentikasi ke dalam aplikasi Anda. Lihat [Mendapatkan kredensial](#) untuk informasi selengkapnya tentang cara mengintegrasikan otentikasi Login with Amazon dengan aplikasi Anda.

Amazon mengeluarkan ID klien OAuth 2.0 untuk profil keamanan baru Anda. Anda dapat menemukan ID klien di tab Pengaturan Web profil keamanan. Masukkan ID Profil Keamanan di bidang ID Aplikasi Login with Amazon iDP di kumpulan identitas Anda.

**Note**

Anda memasukkan ID Profil Keamanan di bidang ID Aplikasi Login with Amazon iDP di kumpulan identitas Anda. Ini berbeda dari kumpulan pengguna, yang menggunakan ID klien.

## Konfigurasi penyedia eksternal di konsol Amazon Cognito

Untuk menambahkan Login with Amazon identity provider (IDP)

1. Pilih kumpulan Identitas dari konsol [Amazon Cognito](#). Pilih kumpulan identitas.
2. Pilih tab Akses pengguna.
3. Pilih Tambahkan penyedia identitas.
4. Pilih Login with Amazon.
5. Masukkan ID Aplikasi proyek OAuth yang Anda buat saat [Login with Amazon](#). Untuk informasi selengkapnya, lihat [Login with Amazon Documentation](#).
6. Untuk menyetel peran yang diminta Amazon Cognito saat mengeluarkan kredensial kepada pengguna yang telah mengautentikasi dengan penyedia ini, konfigurasi setelan Peran.
  - Anda dapat menetapkan pengguna dari IDP tersebut peran Default yang Anda atur saat mengonfigurasi peran Terautentikasi, atau Anda dapat Memilih peran dengan aturan.
    - i. Jika Anda memilih Pilih peran dengan aturan, masukkan Klaim sumber dari autentikasi pengguna Anda, Operator yang ingin Anda bandingkan dengan klaim, Nilai yang akan menyebabkan kecocokan dengan pilihan peran ini, dan Peran yang ingin Anda tetapkan saat penetapan Peran cocok. Pilih Tambahkan yang lain untuk membuat aturan tambahan berdasarkan kondisi yang berbeda.
    - ii. Pilih Resolusi Peran. Jika klaim pengguna tidak sesuai dengan aturan, Anda dapat menolak kredensial atau mengeluarkan kredensial untuk peran yang Diautentikasi.
7. Untuk mengubah tag utama yang ditetapkan Amazon Cognito saat mengeluarkan kredensial kepada pengguna yang telah diautentikasi dengan penyedia ini, konfigurasi Atribut untuk kontrol akses.
  - a. Untuk tidak menerapkan tag utama, pilih Tidak aktif.
  - b. Untuk menerapkan tag utama berdasarkan sub dan aud klaim, pilih Gunakan pemetaan default.



- c. Untuk membuat skema atribut kustom Anda sendiri ke tag utama, pilih Gunakan pemetaan khusus. Kemudian masukkan kunci Tag yang ingin Anda sumber dari setiap Klaim yang ingin Anda wakili dalam tag.
8. Pilih Simpan perubahan.

## Menggunakan Login with Amazon: Android

Setelah Anda mengautentikasi login Amazon, Anda dapat meneruskan token ke penyedia kredensial Amazon Cognito dalam metode onSuccess antarmuka. TokenListener Kodenya terlihat seperti ini:

```
@Override
public void onSuccess(Bundle response) {
    String token = response.getString(AuthzConstants.BUNDLE_KEY.TOKEN.val);
    Map<String, String> logins = new HashMap<String, String>();
    logins.put("www.amazon.com", token);
    credentialsProvider.setLogins(logins);
}
```

## Menggunakan Login with Amazon: iOS - Objective-C

Setelah Anda mengautentikasi login Amazon, Anda dapat meneruskan token ke penyedia kredensial Amazon Cognito dalam requestDidSucceed metode AMZN: AccessTokenDelegate

```
- (void)requestDidSucceed:(APIResult \*)apiResult {
    if (apiResult.api == kAPIAuthorizeUser) {
        [AIMobileLib getAccessTokenForScopes:[NSArray arrayWithObject:@"profile"]
withOverrideParams:nil delegate:self];
    }
    else if (apiResult.api == kAPIGetAccessToken) {
        credentialsProvider.logins = @{ @(AWSCognitoLoginProviderKeyLoginWithAmazon):
apiResult.result };
    }
}
```

## Menggunakan Login with Amazon: iOS - Swift

Setelah Anda mengautentikasi login Amazon, Anda dapat meneruskan token ke penyedia kredensial Amazon Cognito dengan metode: requestDidSucceed AMZNAccessTokenDelegate

```
func requestDidSucceed(apiResult: APIResult!) {
```

```
if apiResult.api == API.AuthorizeUser {
    AIMobileLib.getAccessTokenForScopes(["profile"], withOverrideParams: nil,
delegate: self)
} else if apiResult.api == API.GetAccessToken {
    credentialsProvider.logins =
[AWSCognitoLoginProviderKey.LoginWithAmazon.rawValue: apiResult.result]
}
}
```

## Gunakan Login with Amazon: JavaScript

Setelah pengguna mengautentikasi dengan Login with Amazon dan diarahkan kembali ke situs web Anda, `access_token` Login with Amazon disediakan dalam string kueri. Berikan token itu ke dalam peta login kredensial.

```
AWS.config.credentials = new AWS.CognitoIdentityCredentials({
  IdentityPoolId: 'IDENTITY_POOL_ID',
  Logins: {
    'www.amazon.com': 'Amazon Access Token'
  }
});
```

## Menggunakan Login with Amazon: Xamarin

### Xamarin untuk Android

```
AmazonAuthorizationManager manager = new AmazonAuthorizationManager(this,
Bundle.Empty);

var tokenListener = new APIListener {
  Success = response => {
    // Get the auth token
    var token = response.GetString(AuthzConstants.BUNDLE_KEY.Token.Val);
    credentials.AddLogin("www.amazon.com", token);
  }
};

// Try and get existing login
manager.GetToken(new[] {
  "profile"
}, tokenListener);
```

## Xamarin untuk iOS

Di AppDelegate.cs, masukkan berikut ini:

```
public override bool OpenUrl (UIApplication application, NSURL url, string
    sourceApplication, NSObject annotation)
{
    // Pass on the url to the SDK to parse authorization code from the url
    bool isValidRedirectSignInURL = AIMobileLib.HandleOpenUrl (url, sourceApplication);
    if(!isValidRedirectSignInURL)
        return false;

    // App may also want to handle url
    return true;
}
```

Kemudian, di ViewController.cs, lakukan hal berikut:

```
public override void ViewDidLoad ()
{
    base.LoadView ();

    // Here we create the Amazon Login Button
    btnLogin = UIButton.FromType (UIButtonType.RoundedRect);
    btnLogin.Frame = new RectangleF (55, 206, 209, 48);
    btnLogin.SetTitle ("Login using Amazon", UIControlState.Normal);
    btnLogin.TouchUpInside += (sender, e) => {
        AIMobileLib.AuthorizeUser (new [] { "profile"}, new AMZNAuthorizationDelegate
    ());
    };
    View.AddSubview (btnLogin);
}

// Class that handles Authentication Success/Failure
public class AMZNAuthorizationDelegate : AIAuthenticationDelegate
{
    public override void RequestDidSucceed(ApiResult apiResult)
    {
        // Your code after the user authorizes application for requested scopes
        var token = apiResult["access_token"];
        credentials.AddLogin("www.amazon.com", token);
    }
}
```

```
public override void RequestDidFail(ApiError errorResponse)
{
    // Your code when the authorization fails
    InvokeOnMainThread(() => new UIAlertView("User Authorization Failed",
errorResponse.Error.Message, null, "Ok", null).Show());
}
}
```

## Menyiapkan Google sebagai kumpulan identitas iDP

Amazon Cognito mengintegrasikan dengan Google untuk menyediakan autentikasi federasi untuk pengguna aplikasi seluler Anda. Bagian ini menjelaskan cara mendaftar dan mengatur aplikasi Anda dengan Google sebagai IDP.

### Android

#### Note

Jika aplikasi Anda menggunakan Google dan tersedia di beberapa platform seluler, Anda harus mengonfigurasinya sebagai Penyedia [OpenID Connect](#). Tambahkan semua ID klien yang dibuat sebagai nilai audiens tambahan untuk integrasi yang lebih baik. Untuk mempelajari lebih lanjut tentang model identitas lintas klien Google, lihat [Identitas lintas klien](#).

### Menyiapkan Google

Untuk mengaktifkan Google Sign-in untuk Android, buat project konsol Google Developers untuk aplikasi Anda.

1. Buka [Konsol Developer Google](#) dan buat proyek baru.
2. Pilih API & Layanan, lalu layar persetujuan OAuth. Sesuaikan informasi yang ditampilkan Google kepada pengguna Anda saat Google meminta persetujuan mereka untuk membagikan data profil mereka dengan aplikasi Anda.
3. Pilih Kredensial, lalu Buat kredensial. Pilih ID klien OAuth. Pilih Android sebagai tipe Aplikasi. Buat ID klien terpisah untuk setiap platform tempat Anda mengembangkan aplikasi.
4. Dari Kredensial, pilih Kelola akun layanan. Pilih Buat akun layanan. Masukkan detail akun layanan Anda, lalu pilih Buat dan lanjutkan.
5. Berikan akses akun layanan ke proyek Anda. Berikan pengguna akses ke akun layanan sesuai kebutuhan aplikasi Anda.

6. Pilih akun layanan baru Anda, pilih tab Tombol, dan tombol Tambah. Buat dan unduh kunci JSON baru.

Untuk informasi selengkapnya tentang cara menggunakan konsol Google Developers, lihat [Membuat dan mengelola project](#) di dokumentasi Google Cloud.

Untuk informasi selengkapnya tentang cara mengintegrasikan Google ke dalam aplikasi Android, lihat [Mengautentikasi pengguna dengan Masuk dengan Google](#) di dokumentasi Google Identity.

Untuk menambahkan penyedia identitas Google (iDP)

1. Pilih kumpulan Identitas dari konsol [Amazon Cognito](#). Pilih kumpulan identitas.
2. Pilih tab Akses pengguna.
3. Pilih Tambahkan penyedia identitas.
4. Pilih Google.
5. Masukkan ID Klien proyek OAuth yang Anda buat di [Google Cloud Platform](#). Untuk informasi selengkapnya, lihat [Menyiapkan OAuth 2.0](#) di Bantuan Google Cloud Platform Console.
6. Untuk menyetel peran yang diminta Amazon Cognito saat mengeluarkan kredensial kepada pengguna yang telah mengautentikasi dengan penyedia ini, konfigurasi setelan Peran.
  - Anda dapat menetapkan pengguna dari IDP tersebut peran Default yang Anda atur saat mengonfigurasi peran Terautentikasi, atau Anda dapat Memilih peran dengan aturan.
    - i. Jika Anda memilih Pilih peran dengan aturan, masukkan Klaim sumber dari autentikasi pengguna Anda, Operator yang ingin Anda bandingkan dengan klaim, Nilai yang akan menyebabkan kecocokan dengan pilihan peran ini, dan Peran yang ingin Anda tetapkan saat penetapan Peran cocok. Pilih Tambahkan yang lain untuk membuat aturan tambahan berdasarkan kondisi yang berbeda.
    - ii. Pilih Resolusi Peran. Jika klaim pengguna tidak sesuai dengan aturan, Anda dapat menolak kredensial atau mengeluarkan kredensial untuk peran yang Diautentikasi.
7. Untuk mengubah tag utama yang ditetapkan Amazon Cognito saat mengeluarkan kredensial kepada pengguna yang telah diautentikasi dengan penyedia ini, konfigurasi Atribut untuk kontrol akses.
  - a. Untuk tidak menerapkan tag utama, pilih Tidak aktif.
  - b. Untuk menerapkan tag utama berdasarkan sub dan aud klaim, pilih Gunakan pemetaan default.

- c. Untuk membuat skema atribut kustom Anda sendiri ke tag utama, pilih Gunakan pemetaan khusus. Kemudian masukkan kunci Tag yang ingin Anda sumber dari setiap Klaim yang ingin Anda wakili dalam tag.
8. Pilih Simpan perubahan.

## Gunakan Google

Untuk mengaktifkan login dengan Google di aplikasi Anda, ikuti instruksi dalam [dokumentasi Google untuk Android](#). Saat pengguna masuk, mereka meminta token otentikasi OpenID Connect dari Google. Amazon Cognito kemudian menggunakan token untuk mengautentikasi pengguna dan menghasilkan pengenal unik.

Kode contoh berikut menunjukkan cara mengambil token otentikasi dari layanan Google Play:

```
GooglePlayServicesUtil.isGooglePlayServicesAvailable(getApplicationContext());
AccountManager am = AccountManager.get(this);
Account[] accounts = am.getAccountsByType(GoogleAuthUtil.GOOGLE_ACCOUNT_TYPE);
String token = GoogleAuthUtil.getToken(getApplicationContext(), accounts[0].name,
    "audience:server:client_id:YOUR_GOOGLE_CLIENT_ID");
Map<String, String> logins = new HashMap<String, String>();
logins.put("accounts.google.com", token);
credentialsProvider.setLogins(logins);
```

## iOS - Objective-C

### Note

Jika aplikasi Anda menggunakan Google dan tersedia di beberapa platform seluler, konfigurasi Google sebagai Penyedia [OpenID Connect](#). Tambahkan semua ID klien yang dibuat sebagai nilai audiens tambahan untuk integrasi yang lebih baik. Untuk mempelajari lebih lanjut tentang model identitas lintas klien Google, lihat [Identitas lintas klien](#).

## Menyiapkan Google

Untuk mengaktifkan Google Sign-in untuk iOS, buat project konsol Google Developers untuk aplikasi Anda.

1. Buka [Konsol Developer Google](#) dan buat proyek baru.

2. Pilih API & Layanan, lalu layar persetujuan OAuth. Sesuaikan informasi yang ditampilkan Google kepada pengguna Anda saat Google meminta persetujuan mereka untuk membagikan data profil mereka dengan aplikasi Anda.
3. Pilih Kredensial, lalu Buat kredensial. Pilih ID klien OAuth. Pilih iOS sebagai tipe Aplikasi. Buat ID klien terpisah untuk setiap platform tempat Anda mengembangkan aplikasi.
4. Dari Kredensial, pilih Kelola akun layanan. Pilih Buat akun layanan. Masukkan detail akun layanan Anda, lalu pilih Buat dan lanjutkan.
5. Berikan akses akun layanan ke proyek Anda. Berikan pengguna akses ke akun layanan sesuai kebutuhan aplikasi Anda.
6. Pilih akun layanan baru Anda. Pilih tab Keys, dan Add key. Buat dan unduh kunci JSON baru.

Untuk informasi selengkapnya tentang cara menggunakan konsol Google Developers, lihat [Membuat dan mengelola project](#) di dokumentasi Google Cloud.

Untuk informasi selengkapnya tentang cara mengintegrasikan Google ke dalam aplikasi iOS, lihat [Google Sign-In untuk iOS](#) di dokumentasi Google Identity.

Untuk menambahkan penyedia identitas Google (iDP)

1. Pilih kumpulan Identitas dari konsol [Amazon Cognito](#). Pilih kumpulan identitas.
2. Pilih tab Akses pengguna.
3. Pilih Tambahkan penyedia identitas.
4. Pilih Google.
5. Masukkan ID Klien proyek OAuth yang Anda buat di [Google Cloud Platform](#). Untuk informasi selengkapnya, lihat [Menyiapkan OAuth 2.0](#) di Bantuan Google Cloud Platform Console.
6. Untuk menyetel peran yang diminta Amazon Cognito saat mengeluarkan kredensial kepada pengguna yang telah mengautentikasi dengan penyedia ini, konfigurasi setelan Peran.
  - Anda dapat menetapkan pengguna dari IDP tersebut peran Default yang Anda atur saat mengonfigurasi peran Terotentikasi, atau Anda dapat Memilih peran dengan aturan.
    - i. Jika Anda memilih Pilih peran dengan aturan, masukkan Klaim sumber dari autentikasi pengguna Anda, Operator yang ingin Anda bandingkan dengan klaim, Nilai yang akan menyebabkan kecocokan dengan pilihan peran ini, dan Peran yang ingin Anda tetapkan saat penetapan Peran cocok. Pilih Tambahkan yang lain untuk membuat aturan tambahan berdasarkan kondisi yang berbeda.

- ii. Pilih Resolusi Peran. Jika klaim pengguna tidak sesuai dengan aturan, Anda dapat menolak kredensial atau mengeluarkan kredensial untuk peran yang Diautentikasi.
7. Untuk mengubah tag utama yang ditetapkan Amazon Cognito saat mengeluarkan kredensial kepada pengguna yang telah diautentikasi dengan penyedia ini, konfigurasi Atribut untuk kontrol akses.
  - a. Untuk tidak menerapkan tag utama, pilih Tidak aktif.
  - b. Untuk menerapkan tag utama berdasarkan sub dan aud klaim, pilih Gunakan pemetaan default.
  - c. Untuk membuat skema atribut kustom Anda sendiri ke tag utama, pilih Gunakan pemetaan khusus. Kemudian masukkan kunci Tag yang ingin Anda sumber dari setiap Klaim yang ingin Anda wakili dalam tag.
8. Pilih Simpan perubahan.

## Gunakan Google

Untuk mengaktifkan login dengan Google di aplikasi Anda, ikuti [Dokumentasi Google untuk iOS](#). Autentikasi yang berhasil menghasilkan token autentikasi OpenID Connect, yang digunakan Amazon Cognito untuk mengautentikasi pengguna dan menghasilkan pengidentifikasi unik.

Autentikasi yang berhasil menghasilkan objek `GTM0Auth2Authentication`, yang berisi `id_token`, yang digunakan Amazon Cognito untuk mengautentikasi pengguna dan menghasilkan pengidentifikasi unik:

```
- (void)finishedWithAuth: (GTM0Auth2Authentication *)auth error: (NSError *) error {
    NSString *idToken = [auth.parameters objectForKey:@"id_token"];
    credentialsProvider.logins = @{ @(AWSCognitoLoginProviderKeyGoogle): idToken };
}
```

## iOS - Swift

### Note

Jika aplikasi Anda menggunakan Google dan tersedia di beberapa platform seluler, konfigurasi Google sebagai Penyedia [OpenID Connect](#). Tambahkan semua ID klien yang dibuat sebagai nilai audiens tambahan untuk integrasi yang lebih baik. Untuk mempelajari lebih lanjut tentang model identitas lintas klien Google, lihat [Identitas lintas klien](#).



## Menyiapkan Google

Untuk mengaktifkan Google Sign-in untuk iOS, buat project konsol Google Developers untuk aplikasi Anda.

1. Buka [Konsol Developer Google](#) dan buat proyek baru.
2. Pilih API & Layanan, lalu layar persetujuan OAuth. Sesuaikan informasi yang ditampilkan Google kepada pengguna Anda saat Google meminta persetujuan mereka untuk membagikan data profil mereka dengan aplikasi Anda.
3. Pilih Kredensial, lalu Buat kredensial. Pilih ID klien OAuth. Pilih iOS sebagai tipe Aplikasi. Buat ID klien terpisah untuk setiap platform tempat Anda mengembangkan aplikasi.
4. Dari Kredensial, pilih Kelola akun layanan. Pilih Buat akun layanan. Masukkan detail akun layanan Anda, lalu pilih Buat dan lanjutkan.
5. Berikan akses akun layanan ke proyek Anda. Berikan pengguna akses ke akun layanan sesuai kebutuhan aplikasi Anda.
6. Pilih akun layanan baru Anda, pilih tab Tombol, dan tombol Tambah. Buat dan unduh kunci JSON baru.

Untuk informasi selengkapnya tentang cara menggunakan konsol Google Developers, lihat [Membuat dan mengelola project](#) di dokumentasi Google Cloud.

Untuk informasi selengkapnya tentang cara mengintegrasikan Google ke dalam aplikasi iOS, lihat [Google Sign-In untuk iOS](#) di dokumentasi Google Identity.

Pilih Mengelola Kolam Identitas dari [Halaman beranda konsol Amazon Cognito](#):

Mengonfigurasi penyedia eksternal di Amazon Cognito Console

1. Pilih nama kumpulan identitas tempat Anda ingin mengaktifkan Google sebagai penyedia eksternal. Halaman Dasbor untuk kolam identitas Anda akan muncul.
2. Di sudut kanan atas halaman Dasbor, pilih Edit kolam identitas. Halaman Edit kolam identitas akan muncul.
3. Gulir ke bawah dan pilih Penyedia otentikasi untuk memperluas bagian.
4. Pilih tab Google.
5. Pilih Buka kunci.
6. Masukkan ID Klien Google yang Anda peroleh dari Google, lalu pilih Simpan Perubahan.

## Gunakan Google

Untuk mengaktifkan login dengan Google di aplikasi Anda, ikuti [Dokumentasi Google untuk iOS](#). Otentikasi yang berhasil menghasilkan token otentikasi OpenID Connect yang digunakan Amazon Cognito untuk mengotentikasi pengguna dan menghasilkan pengenal unik.

Otentikasi yang berhasil menghasilkan `GTMOAuth2Authentication` objek yang berisi `fileid_token`. Amazon Cognito menggunakan token ini untuk mengotentikasi pengguna dan menghasilkan pengenal unik:

```
func finishedWithAuth(auth: GTMOAuth2Authentication!, error: NSError!) {
    if error != nil {
        print(error.localizedDescription)
    }
    else {
        let idToken = auth.parameters.objectForKey("id_token")
        credentialsProvider.logins = [AWSCognitoLoginProviderKey.Google.rawValue:
idToken!]
    }
}
```

## JavaScript

### Note

Jika aplikasi Anda menggunakan Google dan tersedia di beberapa platform seluler, Anda harus mengonfigurasi Google sebagai Penyedia [OpenID Connect](#). Tambahkan semua ID klien yang dibuat sebagai nilai audiens tambahan untuk integrasi yang lebih baik. Untuk mempelajari lebih lanjut tentang model identitas lintas klien Google, lihat [Identitas lintas klien](#).

## Menyiapkan Google

Untuk mengaktifkan Google Sign-in untuk aplikasi JavaScript web, buat project konsol Google Developers untuk aplikasi Anda.

1. Buka [Konsol Developer Google](#) dan buat proyek baru.
2. Pilih API & Layanan, lalu layar persetujuan OAuth. Sesuaikan informasi yang ditampilkan Google kepada pengguna Anda saat Google meminta persetujuan mereka untuk membagikan data profil mereka dengan aplikasi Anda.

3. Pilih Kredensial, lalu Buat kredensial. Pilih ID klien OAuth. Pilih aplikasi Web sebagai tipe Aplikasi. Buat ID klien terpisah untuk setiap platform tempat Anda mengembangkan aplikasi.
4. Dari Kredensial, pilih Kelola akun layanan. Pilih Buat akun layanan. Masukkan detail akun layanan Anda, lalu pilih Buat dan lanjutkan.
5. Berikan akses akun layanan ke proyek Anda. Berikan pengguna akses ke akun layanan sesuai kebutuhan aplikasi Anda.
6. Pilih akun layanan baru Anda, pilih tab Tombol, dan tombol Tambah. Buat dan unduh kunci JSON baru.

Untuk informasi selengkapnya tentang cara menggunakan konsol Google Developers, lihat [Membuat dan mengelola project](#) di dokumentasi Google Cloud.

Untuk informasi selengkapnya tentang cara mengintegrasikan Google ke aplikasi web Anda, lihat [Masuk Dengan Google](#) di dokumentasi Identitas Google.

Konfigurasi Penyedia Eksternal di Konsol Amazon Cognito

Untuk menambahkan penyedia identitas Google (iDP)

1. Pilih kumpulan Identitas dari konsol [Amazon Cognito](#). Pilih kumpulan identitas.
2. Pilih tab Akses pengguna.
3. Pilih Tambahkan penyedia identitas.
4. Pilih Google.
5. Masukkan ID Klien proyek OAuth yang Anda buat di [Google Cloud Platform](#). Untuk informasi selengkapnya, lihat [Menyiapkan OAuth 2.0](#) di Bantuan Google Cloud Platform Console.
6. Untuk menyetel peran yang diminta Amazon Cognito saat mengeluarkan kredensial kepada pengguna yang telah mengotentikasi dengan penyedia ini, konfigurasi setelan Peran.
  - Anda dapat menetapkan pengguna dari IDP tersebut peran Default yang Anda atur saat mengonfigurasi peran Terotentikasi, atau Anda dapat Memilih peran dengan aturan.
    - i. Jika Anda memilih Pilih peran dengan aturan, masukkan Klaim sumber dari autentikasi pengguna Anda, Operator yang ingin Anda bandingkan dengan klaim, Nilai yang akan menyebabkan kecocokan dengan pilihan peran ini, dan Peran yang ingin Anda tetapkan saat penetapan Peran cocok. Pilih Tambahkan yang lain untuk membuat aturan tambahan berdasarkan kondisi yang berbeda.

- ii. Pilih Resolusi Peran. Jika klaim pengguna tidak sesuai dengan aturan, Anda dapat menolak kredensial atau mengeluarkan kredensial untuk peran yang Diautentikasi.
7. Untuk mengubah tag utama yang ditetapkan Amazon Cognito saat mengeluarkan kredensial kepada pengguna yang telah diautentikasi dengan penyedia ini, konfigurasi Atribut untuk kontrol akses.
  - a. Untuk tidak menerapkan tag utama, pilih Tidak aktif.
  - b. Untuk menerapkan tag utama berdasarkan sub dan aud klaim, pilih Gunakan pemetaan default.
  - c. Untuk membuat skema atribut kustom Anda sendiri ke tag utama, pilih Gunakan pemetaan khusus. Kemudian masukkan kunci Tag yang ingin Anda sumber dari setiap Klaim yang ingin Anda wakili dalam tag.
8. Pilih Simpan perubahan.

## Gunakan Google

Untuk mengaktifkan login dengan Google di aplikasi Anda, ikuti [Dokumentasi Google untuk Web](#).

Otentikasi yang berhasil menghasilkan objek respons yang berisi `id_token` yang digunakan Amazon Cognito untuk mengautentikasi pengguna dan menghasilkan pengenal unik:

```
function signinCallback(authResult) {
  if (authResult['status']['signed_in']) {

    // Add the Google access token to the Amazon Cognito credentials login map.
    AWS.config.credentials = new AWS.CognitoIdentityCredentials({
      IdentityPoolId: 'IDENTITY_POOL_ID',
      Logins: {
        'accounts.google.com': authResult['id_token']
      }
    });

    // Obtain AWS credentials
    AWS.config.credentials.get(function(){
      // Access AWS resources here.
    });
  }
}
```

## Unity

### Menyiapkan Google

Untuk mengaktifkan Google Sign-in untuk aplikasi Unity, buat project konsol Google Developers untuk aplikasi Anda.

1. Buka [Konsol Developer Google](#) dan buat proyek baru.
2. Pilih API & Layanan, lalu layar persetujuan OAuth. Sesuaikan informasi yang ditampilkan Google kepada pengguna Anda saat Google meminta persetujuan mereka untuk membagikan data profil mereka dengan aplikasi Anda.
3. Pilih Kredensial, lalu Buat kredensial. Pilih ID klien OAuth. Pilih aplikasi Web sebagai tipe Aplikasi. Buat ID klien terpisah untuk setiap platform tempat Anda mengembangkan aplikasi.
4. Untuk Unity, buat ID klien OAuth tambahan untuk Android, dan satu lagi untuk iOS.
5. Dari Kredensial, pilih Kelola akun layanan. Pilih Buat akun layanan. Masukkan detail akun layanan Anda, lalu pilih Buat dan lanjutkan.
6. Berikan akses akun layanan ke proyek Anda. Berikan pengguna akses ke akun layanan sesuai kebutuhan aplikasi Anda.
7. Pilih akun layanan baru Anda, pilih tab Tombol, dan tombol Tambah. Buat dan unduh kunci JSON baru.

Untuk informasi selengkapnya tentang cara menggunakan konsol Google Developers, lihat [Membuat dan mengelola project](#) di dokumentasi Google Cloud.

### Buat Penyedia OpenID di Konsol IAM

1. Buat Penyedia OpenID di Konsol IAM. Untuk informasi tentang cara menyiapkan Penyedia OpenID, lihat Menggunakan Penyedia Identitas [OpenID Connect](#).
2. Ketika diminta untuk URL Penyedia Anda, masukkan "https://accounts.google.com".
3. Ketika diminta untuk memasukkan nilai di bidang Audiens, masukkan salah satu dari tiga ID klien yang Anda buat di langkah sebelumnya.
4. Pilih nama penyedia dan tambahkan dua pemirsa lagi dengan dua ID klien lainnya.

### Konfigurasi Penyedia Eksternal di Konsol Amazon Cognito

Pilih Mengelola Kolam Identitas dari [Halaman beranda konsol Amazon Cognito](#):

## Untuk menambahkan penyedia identitas Google (iDP)

1. Pilih kumpulan Identitas dari konsol [Amazon Cognito](#). Pilih kumpulan identitas.
2. Pilih tab Akses pengguna.
3. Pilih Tambahkan penyedia identitas.
4. Pilih Google.
5. Masukkan ID Klien proyek OAuth yang Anda buat di [Google Cloud Platform](#). Untuk informasi selengkapnya, lihat [Menyiapkan OAuth 2.0](#) di Bantuan Google Cloud Platform Console.
6. Untuk menyetel peran yang diminta Amazon Cognito saat mengeluarkan kredensial kepada pengguna yang telah mengautentikasi dengan penyedia ini, konfigurasi setelan Peran.
  - Anda dapat menetapkan pengguna dari IDP tersebut peran Default yang Anda atur saat mengonfigurasi peran Terautentikasi, atau Anda dapat Memilih peran dengan aturan.
    - i. Jika Anda memilih Pilih peran dengan aturan, masukkan Klaim sumber dari autentikasi pengguna Anda, Operator yang ingin Anda bandingkan dengan klaim, Nilai yang akan menyebabkan kecocokan dengan pilihan peran ini, dan Peran yang ingin Anda tetapkan saat penetapan Peran cocok. Pilih Tambahkan yang lain untuk membuat aturan tambahan berdasarkan kondisi yang berbeda.
    - ii. Pilih Resolusi Peran. Jika klaim pengguna tidak sesuai dengan aturan, Anda dapat menolak kredensial atau mengeluarkan kredensial untuk peran yang Diautentikasi.
7. Untuk mengubah tag utama yang ditetapkan Amazon Cognito saat mengeluarkan kredensial kepada pengguna yang telah diautentikasi dengan penyedia ini, konfigurasi Atribut untuk kontrol akses.
  - a. Untuk tidak menerapkan tag utama, pilih Tidak aktif.
  - b. Untuk menerapkan tag utama berdasarkan sub dan aud klaim, pilih Gunakan pemetaan default.
  - c. Untuk membuat skema atribut kustom Anda sendiri ke tag utama, pilih Gunakan pemetaan khusus. Kemudian masukkan kunci Tag yang ingin Anda sumber dari setiap Klaim yang ingin Anda wakili dalam tag.
8. Pilih Simpan perubahan.

## Instal Plugin Google Unity

1. Tambahkan [Plugin Google Play Games untuk Unity](#) ke proyek Unity Anda.

2. Di Unity, dari menu Windows, gunakan tiga ID untuk platform Android dan iOS untuk mengonfigurasi plugin.

## Gunakan Google

Kode contoh berikut menunjukkan cara mengambil token otentikasi dari layanan Google Play:

```
void Start()
{
    PlayGamesClientConfiguration config = new
    PlayGamesClientConfiguration.Builder().Build();
    PlayGamesPlatform.InitializeInstance(config);
    PlayGamesPlatform.DebugLogEnabled = true;
    PlayGamesPlatform.Activate();
    Social.localUser.Authenticate(GoogleLoginCallback);
}

void GoogleLoginCallback(bool success)
{
    if (success)
    {
        string token = PlayGamesPlatform.Instance.GetIdToken();
        credentials.AddLogin("accounts.google.com", token);
    }
    else
    {
        Debug.LogError("Google login failed. If you are not running in an actual Android/
        iOS device, this is expected.");
    }
}
```

## Xamarin

### Note

Amazon Cognito tidak mendukung Google secara native di platform Xamarin. Integrasi saat ini memerlukan penggunaan tampilan web untuk pergi melalui alur masuk peramban. Untuk mempelajari cara integrasi Google bekerja dengan SDK lain, silakan pilih platform lain.

Untuk mengaktifkan login dengan Google di aplikasi Anda, autentikasi pengguna Anda dan dapatkan token OpenID Connect dari mereka. Amazon Cognito menggunakan token ini untuk menghasilkan pengidentifikasi pengguna unik yang dikaitkan dengan identitas Amazon Cognito. Sayangnya, Google SDK untuk Xamarin tidak memungkinkan Anda untuk mengambil token OpenID Connect, jadi gunakan klien alternatif atau aliran web dalam tampilan web.

Setelah Anda memiliki token, Anda dapat mengaturnya di `CognitoAWSCredentials`:

```
credentials.AddLogin("accounts.google.com", token);
```

### Note

Jika aplikasi Anda menggunakan Google dan tersedia di beberapa platform seluler, Anda harus mengonfigurasi Google sebagai Penyedia [OpenID Connect](#). Tambahkan semua ID klien yang dibuat sebagai nilai audiens tambahan untuk integrasi yang lebih baik. Untuk mempelajari lebih lanjut tentang model identitas lintas klien Google, lihat [Identitas lintas klien](#).

## Menyiapkan Masuk dengan Apple sebagai kumpulan identitas iDP

Amazon Cognito mengintegrasikan dengan Masuk dengan Apple untuk menyediakan autentikasi federasi untuk pengguna aplikasi seluler dan aplikasi web Anda. Bagian ini menjelaskan cara mendaftar dan menyiapkan aplikasi Anda menggunakan Masuk dengan Apple sebagai penyedia identitas (iDP).

Untuk menambahkan Masuk dengan Apple sebagai penyedia otentikasi ke kumpulan identitas, Anda harus menyelesaikan dua prosedur. Pertama, integrasikan Masuk dengan Apple dalam aplikasi, lalu konfigurasi Masuk dengan Apple di kumpulan identitas. Untuk mengetahui up-to-date informasi selengkapnya tentang pengaturan Masuk dengan Apple, lihat [Mengonfigurasi Lingkungan Anda untuk Masuk dengan Apple](#) di dokumentasi Pengembang Apple.

### Mengatur Masuk dengan Apple

Untuk mengonfigurasi Masuk dengan Apple sebagai IDP, daftarkan aplikasi Anda dengan Apple untuk menerima ID klien.


1. Buat sebuah [akun developer dengan Apple](#).
2. [Masuk](#) dengan kredensial Apple Anda.
3. Di panel navigasi sebelah kiri, pilih Sertifikat, ID & Profil.



4. Di panel navigasi sebelah kiri, pilih Pengidentifikasi.
5. Pada halaman Pengidentifikasi, pilih ikon +.
6. Pada halaman Daftarkan Pengidentifikasi Baru, pilih ID Aplikasi, lalu pilih Lanjutkan.
7. Pada halaman Daftarkan ID Aplikasi, lakukan hal berikut:
  - a. Di bawah Deskripsi, ketikkan deskripsi.
  - b. Di bawah ID Paket, ketikkan pengidentifikasi. Catat ID Bundel ini karena Anda memerlukan nilai ini untuk mengonfigurasi Apple sebagai penyedia di kumpulan identitas.
  - c. Pada Kemampuan, pilih Masuk dengan Apple, lalu pilih Edit.
  - d. Pada halaman Masuk dengan Apple: Konfigurasi ID Aplikasi, pilih setelan yang sesuai untuk aplikasi Anda. Lalu, pilih Simpan.
  - e. Pilih Lanjutkan.
8. Pada halaman Konfirmasi ID Aplikasi Anda, pilih Daftarkan.
9. Lanjutkan ke langkah 10 jika Anda ingin mengintegrasikan Masuk dengan Apple dengan aplikasi iOS native. Langkah 11 adalah untuk aplikasi yang ingin Anda integrasikan dengan JS Masuk dengan Apple.
10. Pada halaman Identifier, pilih menu ID Aplikasi, lalu ID Layanan. Pilih ikon +.
11. Pada halaman Daftarkan Pengidentifikasi Baru, pilih ID Layanan, lalu pilih Lanjutkan.
12. Pada halaman Daftarkan ID Layanan, lakukan hal berikut:
  - a. Di bawah Deskripsi, ketikkan deskripsi.
  - b. Di bawah Pengidentifikasi, ketikkan pengidentifikasi. Catat ID layanan karena Anda memerlukan nilai ini untuk mengonfigurasi Apple sebagai penyedia di kumpulan identitas Anda.
  - c. Pilih Masuk dengan Apple, lalu pilih Konfigurasi.
  - d. Pada halaman Konfigurasi Autentikasi Web, pilih ID Aplikasi Utama. Di bawah URL Situs Web, pilih ikon +. Untuk Domain dan Subdomain, masukkan nama domain aplikasi Anda. Di URL Pengembalian, masukkan URL panggilan balik tempat otorisasi mengalihkan pengguna setelah mereka mengautentikasi melalui Masuk dengan Apple.
  - e. Pilih Selanjutnya.
  - f. Pilih Lanjutkan, lalu pilih Daftarkan.
13. Di panel navigasi sebelah kiri, pilih Kunci.
14. Pada halaman Kunci, pilih ikon +.

15. Pada halaman Daftarkan Kunci Baru, lakukan hal berikut:

- a. Di bawah Nama Kunci, ketikkan nama kunci.
- b. Pilih Masuk dengan Apple, lalu pilih Konfigurasi.
- c. Pada halaman Konfigurasi Kunci, pilih ID Aplikasi Utama, lalu pilih Simpan.
- d. Pilih Lanjutkan, lalu pilih Daftarkan.

 Note

Untuk mengintegrasikan Masuk dengan Apple dengan aplikasi iOS native, lihat [Menerapkan Autentikasi Pengguna dengan Masuk dengan Apple](#).

Untuk mengintegrasikan Masuk dengan Apple di platform selain iOS native, lihat [JS Masuk dengan Apple](#).

## Konfigurasi penyedia eksternal di konsol identitas federasi Amazon Cognito

Gunakan prosedur berikut untuk mengonfigurasi penyedia eksternal Anda.

Untuk menambahkan Masuk dengan penyedia identitas Apple (iDP)

1. Pilih kumpulan Identitas dari konsol [Amazon Cognito](#). Pilih kumpulan identitas.
2. Pilih tab Akses pengguna.
3. Pilih Tambahkan penyedia identitas.
4. Pilih Masuk dengan Apple.
5. Masukkan ID Layanan proyek OAuth yang Anda buat dengan [Apple Developer](#). Untuk informasi selengkapnya, lihat [Mengautentikasi pengguna dengan Masuk dengan Apple](#) Masuk dengan Dokumentasi Apple.
6. Untuk menyetel peran yang diminta Amazon Cognito saat mengeluarkan kredensial kepada pengguna yang telah mengautentikasi dengan penyedia ini, konfigurasi setelan Peran.
  - Anda dapat menetapkan pengguna dari IDP tersebut peran Default yang Anda atur saat mengonfigurasi peran Terautentikasi, atau Anda dapat Memilih peran dengan aturan.
    - i. Jika Anda memilih Pilih peran dengan aturan, masukkan Klaim sumber dari autentikasi pengguna Anda, Operator yang ingin Anda bandingkan dengan klaim, Nilai yang akan menyebabkan kecocokan dengan pilihan peran ini, dan Peran yang ingin Anda

- tetapkan saat penetapan Peran cocok. Pilih Tambahkan yang lain untuk membuat aturan tambahan berdasarkan kondisi yang berbeda.
- ii. Pilih Resolusi Peran. Jika klaim pengguna tidak sesuai dengan aturan, Anda dapat menolak kredensial atau mengeluarkan kredensial untuk peran yang Diautentikasi.
7. Untuk mengubah tag utama yang ditetapkan Amazon Cognito saat mengeluarkan kredensial kepada pengguna yang telah diautentikasi dengan penyedia ini, konfigurasi Atribut untuk kontrol akses.
- a. Untuk tidak menerapkan tag utama, pilih Tidak aktif.
  - b. Untuk menerapkan tag utama berdasarkan sub dan aud klaim, pilih Gunakan pemetaan default.
  - c. Untuk membuat skema atribut kustom Anda sendiri ke tag utama, pilih Gunakan pemetaan khusus. Kemudian masukkan kunci Tag yang ingin Anda sumber dari setiap Klaim yang ingin Anda wakili dalam tag.
8. Pilih Simpan perubahan.

## Masuk dengan Apple sebagai penyedia dalam contoh CLI identitas federasi Amazon Cognito

Contoh ini membuat kumpulan identitas bernama `MyIdentityPool` dengan Masuk dengan Apple sebagai iDP.

```
aws cognito-identity create-identity-pool --identity-pool-name MyIdentityPool --supported-login-providers appleid.apple.com="sameple.apple.clientid"
```

Untuk informasi selengkapnya, lihat [Membuat kolam identitas](#)

Menghasilkan sebuah ID identitas Amazon Cognito

Contoh ini menghasilkan (atau mengambil) ID Amazon Cognito. Ini adalah API publik sehingga Anda tidak perlu kredensial apa pun untuk memanggil API ini.

```
aws cognito-identity get-id --identity-pool-id SampleIdentityPoolId --logins appleid.apple.com="SignInWithAppleIdToken"
```

Untuk informasi lebih lanjut, lihat [get-id](#).

Mendapatkan kredensial untuk ID identitas Amazon Cognito

Contoh ini mengembalikan kredensial untuk ID identitas yang diberikan dan Masuk dengan login Apple. Ini adalah API publik sehingga Anda tidak perlu kredensial apa pun untuk memanggil API ini.

```
aws cognito-identity get-credentials-for-identity --identity-id
SampleIdentityId --logins appleid.apple.com="SignInWithAppleIdToken"
```

Untuk informasi lebih lanjut, lihat [get-credentials-for-identity](#)

## Menggunakan Masuk dengan Apple: Android

Apple tidak menyediakan SDK yang mendukung Masuk dengan Apple untuk Android. Anda dapat menggunakan alur web dalam tampilan web sebagai gantinya.

- Untuk mengonfigurasi Masuk dengan Apple di aplikasi anda, ikuti [Mengonfigurasi halaman Web Anda untuk Masuk dengan Apple](#) dalam dokumentasi Apple.
- Untuk menambahkan tombol Masuk dengan Apple ke antarmuka pengguna Android Anda, ikuti [Menampilkan dan Mengonfigurasi Tombol Masuk dengan Apple](#) dalam dokumentasi Apple.
- Untuk mengautentikasi pengguna dengan aman dengan Masuk dengan Apple, ikuti [Mengautentikasi Pengguna dengan Masuk dengan Apple di dokumentasi Apple](#).

Masuk dengan Apple menggunakan objek sesi untuk melacak keadaannya. Amazon Cognito menggunakan token ID dari objek sesi ini untuk mengautentikasi pengguna, menghasilkan pengenal unik, dan, jika diperlukan, memberikan pengguna akses ke sumber daya lain. AWS

```
@Override
public void onSuccess(Bundle response) {
    String token = response.getString("id_token");
    Map<String, String> logins = new HashMap<String, String>();
    logins.put("appleid.apple.com", token);
    credentialsProvider.setLogins(logins);
}
```

## Menggunakan Masuk dengan Apple: iOS - Objective-C

Apple menyediakan dukungan SDK untuk Masuk dengan Apple di aplikasi iOS native. Untuk menerapkan autentikasi pengguna dengan Masuk dengan Apple di perangkat iOS native, ikuti [Menerapkan Autentikasi Pengguna dengan Masuk dengan Apple](#) dalam dokumentasi Apple.

Amazon Cognito menggunakan token ID untuk mengautentikasi pengguna, menghasilkan pengenal unik, dan, jika diperlukan, memberikan pengguna akses ke sumber daya lain. AWS

```
(void)finishedWithAuth: (ASAuthorizationAppleIDCredential *)auth error: (NSError *)
error {
    NSString *idToken = [ASAuthorizationAppleIDCredential
objectForKey:@"identityToken"];
    credentialsProvider.logins = @{ "appleid.apple.com": idToken };
}
```

## Menggunakan Masuk dengan Apple: iOS - Swift

Apple menyediakan dukungan SDK untuk Masuk dengan Apple di aplikasi iOS native. Untuk menerapkan autentikasi pengguna dengan Masuk dengan Apple di perangkat iOS native, ikuti [Menerapkan Autentikasi Pengguna dengan Masuk dengan Apple](#) dalam dokumentasi Apple.

Amazon Cognito menggunakan token ID untuk mengautentikasi pengguna, menghasilkan pengenal unik, dan, jika diperlukan, memberikan pengguna akses ke sumber daya lain. AWS

Untuk informasi selengkapnya tentang cara mengatur Masuk dengan Apple di iOS, lihat [Mengatur Masuk dengan Apple](#)

```
func finishedWithAuth(auth: ASAuthorizationAppleIDCredential!, error: NSError!) {
    if error != nil {
        print(error.localizedDescription)
    }
    else {
        let idToken = auth.identityToken,
        credentialsProvider.logins = ["appleid.apple.com": idToken!]
    }
}
```

## Gunakan Masuk dengan Apple: JavaScript

Apple tidak menyediakan SDK yang mendukung Masuk dengan Apple untuk JavaScript. Anda dapat menggunakan alur web dalam tampilan web sebagai gantinya.

- Untuk mengonfigurasi Masuk dengan Apple di aplikasi anda, ikuti [Mengonfigurasi halaman Web Anda untuk Masuk dengan Apple](#) dalam dokumentasi Apple.
- Untuk menambahkan tombol Masuk dengan Apple ke antarmuka JavaScript pengguna, ikuti [Menampilkan dan Mengonfigurasi Tombol Masuk dengan Apple](#) di dokumentasi Apple.
- Untuk mengautentikasi pengguna dengan aman melalui Masuk dengan Apple, ikuti [Mengonfigurasi halaman Web Anda untuk Masuk dengan Apple di dokumentasi Apple](#).

Masuk dengan Apple menggunakan objek sesi untuk melacak keadaannya. Amazon Cognito menggunakan token ID dari objek sesi ini untuk mengautentikasi pengguna, menghasilkan pengenal unik, dan, jika diperlukan, memberikan pengguna akses ke sumber daya lain. AWS

```
function signinCallback(authResult) {
    // Add the apple's id token to the Amazon Cognito credentials login map.
    AWS.config.credentials = new AWS.CognitoIdentityCredentials({
        IdentityPoolId: 'IDENTITY_POOL_ID',
        Logins: {
            'appleid.apple.com': authResult['id_token']
        }
    });

    // Obtain AWS credentials
    AWS.config.credentials.get(function(){
        // Access AWS resources here.
    });
}
```

## Menggunakan Masuk dengan Apple: Xamarin

Kami tidak memiliki SDK yang mendukung Masuk dengan Apple untuk Xamarin. Anda dapat menggunakan alur web dalam tampilan web sebagai gantinya.

- Untuk mengonfigurasi Masuk dengan Apple di aplikasi anda, ikuti [Mengonfigurasi halaman Web Anda untuk Masuk dengan Apple](#) dalam dokumentasi Apple.
- Untuk menambahkan tombol Masuk dengan Apple ke antarmuka pengguna Xamarin Anda, ikuti [Menampilkan dan Mengonfigurasi Tombol Masuk dengan Apple](#) dalam dokumentasi Apple.
- Untuk mengautentikasi pengguna dengan aman melalui Masuk dengan Apple, ikuti [Mengonfigurasi halaman Web Anda untuk Masuk dengan Apple di dokumentasi Apple](#).

Masuk dengan Apple menggunakan objek sesi untuk melacak keadaannya. Amazon Cognito menggunakan token ID dari objek sesi ini untuk mengautentikasi pengguna, menghasilkan pengenal unik, dan, jika diperlukan, memberikan pengguna akses ke sumber daya lain. AWS

Setelah Anda memiliki token, Anda dapat mengaturnya di `CognitoAWSCredentials`:

```
credentials.AddLogin("appleid.apple.com", token);
```

## Menyiapkan penyedia OIDC sebagai idP kumpulan identitas

[OpenID Connect](#) adalah standar terbuka untuk otentikasi yang didukung oleh sejumlah penyedia login. Amazon Cognito mendukung Anda untuk menautkan identitas dengan penyedia OpenID Connect yang Anda konfigurasi. [AWS Identity and Access Management](#)

### Menambahkan penyedia OpenID Connect

Untuk informasi tentang cara membuat penyedia OpenID Connect, lihat Membuat penyedia [identitas OpenID Connect \(OIDC\)](#) di Panduan Pengguna.AWS Identity and Access Management

### Mengaitkan penyedia dengan Amazon Cognito

Untuk menambahkan penyedia identitas OIDC (iDP)

1. Pilih kumpulan Identitas dari konsol [Amazon Cognito](#). Pilih kumpulan identitas.
2. Pilih tab Akses pengguna.
3. Pilih Tambahkan penyedia identitas.
4. Pilih OpenID Connect (OIDC).
5. Pilih penyedia identitas OIDC dari IAM IdPs di Anda. Akun AWS Jika Anda ingin menambahkan penyedia SAMP baru, pilih Buat penyedia baru untuk menavigasi ke konsol IAM.
6. Untuk menyetel peran yang diminta Amazon Cognito saat mengeluarkan kredensial kepada pengguna yang telah mengautentikasi dengan penyedia ini, konfigurasi setelan Peran.
  - Anda dapat menetapkan pengguna dari IDP tersebut peran Default yang Anda atur saat mengonfigurasi peran Terautentikasi, atau Anda dapat Memilih peran dengan aturan.
    - i. Jika Anda memilih Pilih peran dengan aturan, masukkan Klaim sumber dari autentikasi pengguna Anda, Operator yang ingin Anda bandingkan dengan klaim, Nilai yang akan menyebabkan kecocokan dengan pilihan peran ini, dan Peran yang ingin Anda tetapkan saat penetapan Peran cocok. Pilih Tambahkan yang lain untuk membuat aturan tambahan berdasarkan kondisi yang berbeda.
    - ii. Pilih Resolusi Peran. Jika klaim pengguna tidak sesuai dengan aturan, Anda dapat menolak kredensial atau mengeluarkan kredensial untuk peran yang Diautentikasi.
7. Untuk mengubah tag utama yang ditetapkan Amazon Cognito saat mengeluarkan kredensial kepada pengguna yang telah diautentikasi dengan penyedia ini, konfigurasi Atribut untuk kontrol akses.

- a. Untuk tidak menerapkan tag utama, pilih Tidak aktif.
  - b. Untuk menerapkan tag utama berdasarkan sub dan aud klaim, pilih Gunakan pemetaan default.
  - c. Untuk membuat skema atribut kustom Anda sendiri ke tag utama, pilih Gunakan pemetaan khusus. Kemudian masukkan kunci Tag yang ingin Anda sumber dari setiap Klaim yang ingin Anda wakili dalam tag.
8. Pilih Simpan perubahan.

Anda dapat mengaitkan beberapa penyedia OpenID Connect dengan kolam identitas tunggal.

### Menggunakan OpenID Connect

Lihat dokumentasi penyedia Anda untuk cara login dan menerima token ID.

Setelah Anda memiliki token, tambahkan token ke peta login. Gunakan URI penyedia Anda sebagai kuncinya.

### Memvalidasi token OpenID Connect

Saat pertama kali berintegrasi dengan Amazon Cognito, Anda mungkin menerima pengecualian `InvalidToken`. Penting untuk memahami bagaimana Amazon Cognito memvalidasi token OpenID Connect (OIDC).

#### Note

Seperti yang ditentukan di sini (<https://tools.ietf.org/html/rfc7523>), Amazon Cognito memberikan masa tenggang 5 menit untuk menangani kemiringan jam antar sistem.

1. `issParameter` harus cocok dengan kunci yang digunakan peta login (seperti `login.provider.com`).
2. Tanda tangan harus valid. Tanda tangan harus dapat diverifikasi melalui kunci publik RSA.
3. Sidik jari kunci publik sertifikat cocok dengan sidik jari yang Anda tetapkan di IAM saat Anda membuat penyedia OIDC Anda.
4. Jika `azp` parameter ada, periksa nilai ini terhadap ID klien yang terdaftar di penyedia OIDC Anda.
5. Jika `azp` parameter tidak ada, periksa `aud` parameter terhadap ID klien yang terdaftar di penyedia OIDC Anda.



Situs web [jwt.io](http://jwt.io) adalah sumber daya berharga yang dapat Anda gunakan untuk memecahkan kode token dan memverifikasi nilai-nilai ini.

## Android

```
Map<String, String> logins = new HashMap<String, String>();
logins.put("login.provider.com", token);
credentialsProvider.setLogins(logins);
```

## iOS - Objective-C

```
credentialsProvider.logins = @{ "login.provider.com": token }
```

## iOS - Swift

Untuk menyediakan token ID OIDC ke Amazon Cognito, terapkan protokol `AWSCognitoIdentityProviderManager`.

Saat Anda menerapkan `logins` metode ini, kembalikan kamus yang berisi nama penyedia OIDC yang Anda konfigurasi. Kamus ini bertindak sebagai kunci, dan token ID saat ini dari pengguna yang diautentikasi bertindak sebagai nilai, seperti yang ditunjukkan pada contoh kode berikut.

```
class OIDCProvider: NSObject, AWSCognitoIdentityProviderManager {
    func logins() -> AWSTask<NSDictionary> {
        let completion = AWSTaskCompletionSource<NSString>()
        getToken(tokenCompletion: completion)
        return completion.task.continueOnSuccessWith { (task) -> AWSTask<NSDictionary>?
        in
            //login.provider.name is the name of the OIDC provider as setup in the
            Amazon Cognito console
            return AWSTask(result:["login.provider.name":task.result!])
        } as! AWSTask<NSDictionary>
    }

    func getToken(tokenCompletion: AWSTaskCompletionSource<NSString>) -> Void {
        //get a valid oidc token from your server, or if you have one that hasn't
        expired cached, return it

        //TODO code to get token from your server
        //...
```

```
        //if error getting token, set error appropriately
        tokenCompletion.set(error:NSError(domain: "OIDC Login", code: -1 , userInfo:
["Unable to get OIDC token" : "Details about your error"]))
        //else
        tokenCompletion.set(result:"result from server id token")
    }
}
```

Saat Anda membuat `instanceAWSCognitoCredentialsProvider`, lewati kelas yang mengimplementasikan `AWSIdentityProviderManager` sebagai nilai `identityProviderManager` dalam konstruktor. Untuk informasi lebih lanjut, buka halaman [AWSCognitoCredentialsProvider](#) referensi dan pilih [initWithRegionJenis:identityPoolId: identityProviderManager](#).

## JavaScript

```
AWS.config.credentials = new AWS.CognitoIdentityCredentials({
  IdentityPoolId: 'IDENTITY_POOL_ID',
  Logins: {
    'login.provider.com': token
  }
});
```

## Unity

```
credentials.AddLogin("login.provider.com", token);
```

## Xamarin

```
credentials.AddLogin("login.provider.com", token);
```

## Menyiapkan penyedia SAMP sebagai idP kumpulan identitas

Amazon Cognito mendukung otentikasi dengan penyedia identitas (IdPs) melalui Security Assertion Markup Language 2.0 (SAMP 2.0). Anda dapat menggunakan idP yang mendukung SAMP dengan Amazon Cognito untuk menyediakan alur orientasi sederhana bagi pengguna Anda. IDP pendukung SAML Anda menentukan peran IAM yang dapat diasumsikan oleh pengguna Anda. Dengan cara ini, pengguna yang berbeda dapat menerima set izin yang berbeda.

## Mengonfigurasi kumpulan identitas Anda untuk IDP SAMP

Langkah-langkah berikut menjelaskan cara mengonfigurasi kumpulan identitas Anda untuk menggunakan IDP berbasis SAML.

### Note

[Sebelum Anda mengonfigurasi kumpulan identitas Anda untuk mendukung penyedia SAMP, pertama-tama konfigurasi IDP SAMP di konsol IAM.](#) Untuk informasi lebih lanjut, lihat [Mengintegrasikan penyedia solusi SAML pihak ketiga dengan AWS](#) di Panduan Pengguna IAM.

Untuk menambahkan penyedia identitas SAMP (IDP)

1. Pilih kumpulan Identitas dari konsol [Amazon Cognito](#). Pilih kumpulan identitas.
2. Pilih tab Akses pengguna.
3. Pilih Tambahkan penyedia identitas.
4. Pilih SAML.
5. Pilih penyedia identitas SAMP dari IAM IdPs di Anda. Akun AWS Jika Anda ingin menambahkan penyedia SAMP baru, pilih Buat penyedia baru untuk menavigasi ke konsol IAM.
6. Untuk menyetel peran yang diminta Amazon Cognito saat mengeluarkan kredensial kepada pengguna yang telah mengautentikasi dengan penyedia ini, konfigurasi setelan Peran.
  - Anda dapat menetapkan pengguna dari IDP tersebut peran Default yang Anda atur saat mengonfigurasi peran Terautentikasi, atau Anda dapat Memilih peran dengan aturan.
    - i. Jika Anda memilih Pilih peran dengan aturan, masukkan Klaim sumber dari autentikasi pengguna Anda, Operator yang ingin Anda bandingkan dengan klaim, Nilai yang akan menyebabkan kecocokan dengan pilihan peran ini, dan Peran yang ingin Anda tetapkan saat penetapan Peran cocok. Pilih Tambahkan yang lain untuk membuat aturan tambahan berdasarkan kondisi yang berbeda.
    - ii. Pilih Resolusi Peran. Jika klaim pengguna tidak sesuai dengan aturan, Anda dapat menolak kredensial atau mengeluarkan kredensial untuk peran yang Diautentikasi.
7. Untuk mengubah tag utama yang ditetapkan Amazon Cognito saat mengeluarkan kredensial kepada pengguna yang telah diautentikasi dengan penyedia ini, konfigurasi Atribut untuk kontrol akses.

- a. Untuk tidak menerapkan tag utama, pilih Tidak aktif.
  - b. Untuk menerapkan tag utama berdasarkan sub dan aud klaim, pilih Gunakan pemetaan default.
  - c. Untuk membuat skema atribut kustom Anda sendiri ke tag utama, pilih Gunakan pemetaan khusus. Kemudian masukkan kunci Tag yang ingin Anda sumber dari setiap Klaim yang ingin Anda wakili dalam tag.
8. Pilih Simpan perubahan.

## Mengkonfigurasi IDP SALL Anda

Setelah Anda membuat penyedia SAMP, konfigurasi IDP SAMP Anda untuk menambahkan kepercayaan pihak yang bergantung antara IDP Anda dan AWS. Dengan banyak IDPs, Anda dapat menentukan URL yang dapat digunakan IDP untuk membaca informasi dan sertifikat pihak yang bergantung dari dokumen XHTML. Untuk AWS, Anda dapat menggunakan <https://signin.aws.amazon.com/static/saml-metadata.xml>. Langkah selanjutnya adalah mengonfigurasi respons pernyataan SAMP dari IDP Anda untuk mengisi klaim yang diperlukan. AWS Untuk detail tentang konfigurasi klaim, lihat [Mengonfigurasi pernyataan SAML untuk respons autentikasi](#).

Jika IDP SALL Anda menyertakan lebih dari satu sertifikat penandatanganan dalam metadata SAMP, saat login, kumpulan pengguna Anda menentukan bahwa pernyataan SAMP valid jika cocok dengan sertifikat apa pun dalam metadata SAMP.

## Menyesuaikan peran pengguna Anda dengan SAMP

Saat Anda menggunakan SAMP dengan Amazon Cognito Identity, Anda dapat menyesuaikan peran untuk pengguna akhir. Amazon Cognito hanya mendukung [aliran yang disempurnakan dengan IDP](#) berbasis SAML. Anda tidak perlu menentukan peran yang diautentikasi atau tidak diautentikasi agar kumpulan identitas menggunakan IDP berbasis SAML. Atribut klaim `https://aws.amazon.com/SAML/Attributes/Role` menentukan satu atau lebih pasangan peran yang dipisahkan koma dan ARN penyedia. Ini adalah peran yang dapat diasumsikan pengguna. Anda dapat mengonfigurasi IDP SAMP untuk mengisi atribut peran berdasarkan informasi atribut pengguna yang tersedia dari IDP. Jika Anda menerima beberapa peran dalam pernyataan SAMP, isi `customRoleArn` parameter opsional saat Anda menelepon `getCredentialsForIdentity` Pengguna mengasumsikan ini `customRoleArn` jika peran cocok dengan satu dalam klaim dalam pernyataan SAMP.

## Mengautentikasi pengguna dengan SAMP iDP

Untuk bergabung dengan IDP berbasis SAML, tentukan URL tempat pengguna memulai login. AWS federasi menggunakan login yang diprakarsai IDP. Di AD FS 2.0, URL mengambil bentuk `https://<fqdn>/adfs/ls/IdpInitiatedSignOn.aspx?loginToRp=urn:amazon:webservices`.

Untuk menambahkan dukungan untuk IDP SAMP Anda di Amazon Cognito, pertama-tama autentikasi pengguna dengan penyedia identitas SAMP Anda dari aplikasi iOS atau Android Anda. Kode yang Anda gunakan untuk mengintegrasikan dan mengautentikasi dengan SAMP iDP khusus untuk penyedia SAMP. Setelah mengautentikasi pengguna, Anda dapat menggunakan Amazon Cognito API untuk memberikan pernyataan SAMP yang dihasilkan ke Identitas Amazon Cognito.

Anda tidak dapat mengulangi, atau memutar ulang, pernyataan SAMP di Logins peta permintaan API kumpulan identitas Anda. Pernyataan SAMP yang diputar ulang memiliki ID pernyataan yang menduplikasi ID permintaan API sebelumnya. [Operasi API yang dapat menerima pernyataan SAMP di Logins peta termasuk GetId, GetCredentialsForIdentityGetOpenIdToken, dan ID.GetOpenTokenForDeveloperIdentity](#) Anda dapat memutar ulang ID pernyataan SAMP satu kali per permintaan API dalam alur otentikasi kumpulan identitas. Misalnya, Anda dapat memberikan pernyataan SAMP yang sama dalam GetId permintaan dan permintaan berikutnya, tetapi tidak dalam GetCredentialsForIdentity permintaan kedua. GetId

## Android

Jika Anda menggunakan Android SDK, Anda dapat mengisi peta login dengan pernyataan SAMP sebagai berikut.

```
Map logins = new HashMap();
logins.put("arn:aws:iam::aws account id:saml-provider/name", "base64 encoded assertion
response");
// Now this should be set to CognitoCachingCredentialsProvider object.
CognitoCachingCredentialsProvider credentialsProvider = new
CognitoCachingCredentialsProvider(context, identity pool id, region);
credentialsProvider.setLogins(logins);
// If SAML assertion contains multiple roles, resolve the role by setting the custom
role
credentialsProvider.setCustomRoleArn("arn:aws:iam::aws account id:role/
customRoleName");
// This should trigger a call to the Amazon Cognito service to get the credentials.
credentialsProvider.getCredentials();
```

## iOS

Jika Anda menggunakan SDK iOS, Anda dapat memberikan pernyataan SAML di `AWSCognitoIdentityProviderManager` sebagai berikut.

```
- (AWSTask<NSDictionary<NSString*,NSString*> *> *) logins {
    //this is hardcoded for simplicity, normally you would asynchronously go to your
    SAML provider
    //get the assertion and return the logins map using a AWSTaskCompletionSource
    return [AWSTask taskWithResult:@{@"arn:aws:iam::aws account id:saml-provider/
name":@"base64 encoded assertion response"}];
}

// If SAML assertion contains multiple roles, resolve the role by setting the custom
role.
// Implementing this is optional if there is only one role.
- (NSString *)customRoleArn {
    return @"arn:aws:iam::accountId:role/customRoleName";
}
```

## Identitas yang diautentikasi pengembang (kumpulan identitas)

Amazon Cognito mendukung identitas yang diautentikasi pengembang, selain federasi identitas web melalui,, dan. [Menyiapkan Facebook sebagai kumpulan identitas IDP](#) [Menyiapkan Google sebagai kumpulan identitas iDP](#) [Menyiapkan Login with Amazon sebagai kumpulan identitas IDP](#) [Menyiapkan Masuk dengan Apple sebagai kumpulan identitas iDP](#) Dengan identitas yang diautentikasi pengembang, Anda dapat mendaftarkan dan mengautentikasi pengguna melalui proses autentikasi yang ada, sambil tetap menggunakan Amazon Cognito untuk menyinkronkan data pengguna dan mengakses sumber daya. AWS Menggunakan identitas yang diautentikasi pengembang melibatkan interaksi antara perangkat pengguna akhir, backend Anda untuk autentikasi, dan Amazon Cognito. Untuk detail selengkapnya, lihat [Memahami Otentikasi Amazon Cognito Bagian 2: Identitas Terautentikasi Pengembang](#) di blog. AWS

## Memahami alur otentikasi

Operasi [GetOpenIdTokenForDeveloperIdentity](#) API dapat memulai otentikasi pengembang untuk otentikasi yang ditingkatkan dan dasar. API ini mengautentikasi permintaan dengan kredensi administratif. `LoginsPeta` adalah nama penyedia pengembang kumpulan identitas seperti `login.mydevprovider` dipasangkan dengan pengenalan kustom.

Contoh:

```
"Logins": {
  "login.mydevprovider": "my developer identifier"
}
```

Otentikasi yang disempurnakan

Panggil operasi [GetCredentialsForIdentity](#) API dengan Logins peta dengan nama `cognito-identity.amazonaws.com` dan nilai token dari `getOpenIdTokenForDeveloperIdentity`.

Contoh:

```
"Logins": {
  "cognito-identity.amazonaws.com": "eyJra12345EXAMPLE"
}
```

`GetCredentialsForIdentity` dengan identitas yang diautentikasi pengembang mengembalikan kredensial sementara untuk peran default yang diautentikasi dari kumpulan identitas.

Otentikasi dasar

Panggil operasi [AssumeRoleWithWebIdentity](#) API dan minta peran IAM apa pun yang memiliki [hubungan kepercayaan yang sesuai yang ditentukan](#). RoleArn Tetapkan nilai `WebIdentityToken` ke token yang diperoleh dari `getOpenIdTokenForDeveloperIdentity`.

Untuk informasi tentang authflow identitas yang diautentikasi pengembang dan perbedaannya dari identitas penyedia eksternal, lihat. [Aliran otentikasi kumpulan identitas \(identitas federasi\)](#)

## Tentukan nama penyedia pengembang dan kaitkan dengan kumpulan identitas

Untuk menggunakan identitas yang diautentikasi pengembang, Anda memerlukan kumpulan identitas yang terkait dengan penyedia pengembang Anda. Untuk melakukannya, ikuti langkah-langkah berikut:

Untuk menambahkan penyedia pengembang khusus

1. Pilih kumpulan Identitas dari konsol [Amazon Cognito](#). Pilih kumpulan identitas.
2. Pilih tab Akses pengguna.
3. Pilih Tambahkan penyedia identitas.

4. Pilih Penyedia pengembang khusus.
5. Masukkan nama penyedia Pengembang. Anda tidak dapat mengubah atau menghapus penyedia pengembang setelah menambahkannya.
6. Pilih Simpan perubahan.

Catatan: Setelah nama penyedia telah diatur, itu tidak dapat diubah.

Untuk arahan tambahan mengenai bekerja dengan Konsol Amazon Cognito, lihat [Menggunakan konsol Amazon Cognito](#).

## Menerapkan penyedia identitas

### Android

Untuk menggunakan identitas yang diautentikasi pengembang, terapkan kelas penyedia identitas Anda sendiri yang diperluas. `AWSAbstractCognitoIdentityProvider` Kelas penyedia identitas Anda harus mengembalikan objek respons yang berisi token sebagai atribut.

Berikut ini adalah contoh dasar dari penyedia identitas.

```
public class DeveloperAuthenticationProvider extends
    AWSAbstractCognitoDeveloperIdentityProvider {

    private static final String developerProvider = "<Developer_provider_name>";

    public DeveloperAuthenticationProvider(String accountId, String identityPoolId,
    Regions region) {
        super(accountId, identityPoolId, region);
        // Initialize any other objects needed here.
    }

    // Return the developer provider name which you choose while setting up the
    // identity pool in the &COG; Console

    @Override
    public String getProviderName() {
        return developerProvider;
    }

    // Use the refresh method to communicate with your backend to get an
    // identityId and token.
```



```
@Override
public String refresh() {

    // Override the existing token
    setToken(null);

    // Get the identityId and token by making a call to your backend
    // (Call to your backend)

    // Call the update method with updated identityId and token to make sure
    // these are ready to be used from Credentials Provider.

    update(identityId, token);
    return token;

}

// If the app has a valid identityId return it, otherwise get a valid
// identityId from your backend.

@Override
public String getIdentityId() {

    // Load the identityId from the cache
    identityId = cachedIdentityId;

    if (identityId == null) {
        // Call to your backend
    } else {
        return identityId;
    }

}
}
```

Untuk menggunakan penyedia identitas ini, Anda harus meneruskannya ke `CognitoCachingCredentialsProvider`. Inilah contohnya:

```
DeveloperAuthenticationProvider developerProvider = new
DeveloperAuthenticationProvider( null, "IDENTITYPOOLID", context, Regions.USEAST1);
CognitoCachingCredentialsProvider credentialsProvider = new
CognitoCachingCredentialsProvider( context, developerProvider, Regions.USEAST1);
```

## iOS - Objektif-C

Untuk menggunakan identitas yang diautentikasi pengembang, terapkan kelas penyedia identitas Anda sendiri yang diperluas. [AWSCognitoCredentialsProviderHelper](#) Kelas penyedia identitas Anda harus mengembalikan objek respons yang berisi token sebagai atribut.

```
@implementation DeveloperAuthenticatedIdentityProvider
/*
 * Use the token method to communicate with your backend to get an
 * identityId and token.
 */

- (AWSTask <NSString*> *) token {
    //Write code to call your backend:
    //Pass username/password to backend or some sort of token to authenticate user
    //If successful, from backend call getOpenIdTokenForDeveloperIdentity with logins
    map
    //containing "your.provider.name":"enduser.username"
    //Return the identity id and token to client
    //You can use AWSTaskCompletionSource to do this asynchronously

    // Set the identity id and return the token
    self.identityId = response.identityId;
    return [AWSTask taskWithResult:response.token];
}

@end
```

Untuk menggunakan penyedia identitas ini, teruskan itu ke `AWSCognitoCredentialsProvider` seperti yang ditunjukkan dalam contoh berikut:

```
DeveloperAuthenticatedIdentityProvider * devAuth =
[[DeveloperAuthenticatedIdentityProvider alloc]
 initWithRegionType:AWSRegionYOUR_IDENTITY_POOL_REGION
                identityPoolId:@"YOUR_IDENTITY_POOL_ID"
                useEnhancedFlow:YES
                identityProviderManager:nil];
AWSCognitoCredentialsProvider *credentialsProvider = [[AWSCognitoCredentialsProvider
 alloc]

 initWithRegionType:AWSRegionYOUR_IDENTITY_POOL_REGION
                identityProvider:devAuth];
```

Jika Anda ingin mendukung identitas yang tidak diautentikasi dan identitas yang diautentikasi pengembang, ganti metode dalam implementasi Anda. `logins` `AWSCognitoCredentialsProviderHelper`

```
- (AWSTask<NSDictionary<NSString *, NSString *> *> *)logins {
    if(/*logic to determine if user is unauthenticated*/) {
        return [AWSTask taskWithResult:nil];
    }else{
        return [super logins];
    }
}
```

Jika Anda ingin mendukung identitas dan penyedia sosial yang diautentikasi pengembang, Anda harus mengelola siapa penyedia saat ini dalam implementasi Anda. `logins` `AWSCognitoCredentialsProviderHelper`

```
- (AWSTask<NSDictionary<NSString *, NSString *> *> *)logins {
    if(/*logic to determine if user is unauthenticated*/) {
        return [AWSTask taskWithResult:nil];
    }else if (/*logic to determine if user is Facebook*/){
        return [AWSTask taskWithResult: @{ AWSIdentityProviderFacebook :
[FBSDKAccessToken currentAccessToken] }];
    }else {
        return [super logins];
    }
}
```

## iOS - cepat

Untuk menggunakan identitas yang diautentikasi pengembang, terapkan kelas penyedia identitas Anda sendiri yang diperluas. [AWSCognitoCredentialsProviderHelper](#) Kelas penyedia identitas Anda harus mengembalikan objek respons yang berisi token sebagai atribut.

```
import AWSCore
/*
 * Use the token method to communicate with your backend to get an
 * identityId and token.
 */
class DeveloperAuthenticatedIdentityProvider : AWSCognitoCredentialsProviderHelper {
    override func token() -> AWSTask<NSString> {
        //Write code to call your backend:
    }
}
```

```

    //pass username/password to backend or some sort of token to authenticate user, if
    successful,
    //from backend call getOpenIdTokenForDeveloperIdentity with logins map containing
    "your.provider.name":"enduser.username"
    //return the identity id and token to client
    //You can use AWSTaskCompletionSource to do this asynchronously

    // Set the identity id and return the token
    self.identityId = resultFromAbove.identityId
    return AWSTask(result: resultFromAbove.token)
}

```

Untuk menggunakan penyedia identitas ini, teruskan itu ke `AWSCognitoCredentialsProvider` seperti yang ditunjukkan dalam contoh berikut:

```

let devAuth =
    DeveloperAuthenticatedIdentityProvider(regionType: .YOUR_IDENTITY_POOL_REGION,
    identityPoolId: "YOUR_IDENTITY_POOL_ID", useEnhancedFlow: true,
    identityProviderManager:nil)
let credentialsProvider =
    AWSCognitoCredentialsProvider(regionType: .YOUR_IDENTITY_POOL_REGION,
    identityProvider:devAuth)
let configuration = AWSServiceConfiguration(region: .YOUR_IDENTITY_POOL_REGION,
    credentialsProvider:credentialsProvider)
AWSServiceManager.default().defaultServiceConfiguration = configuration

```

Jika Anda ingin mendukung identitas yang tidak diautentikasi dan identitas yang diautentikasi pengembang, ganti metode dalam implementasi Anda. `logins` `AWSCognitoCredentialsProviderHelper`

```

override func logins () -> AWSTask<NSDictionary> {
    if(/*logic to determine if user is unauthenticated*/) {
        return AWSTask(result:nil)
    }else {
        return super.logins()
    }
}

```

Jika Anda ingin mendukung identitas dan penyedia sosial yang diautentikasi pengembang, Anda harus mengelola siapa penyedia saat ini dalam implementasi Anda. `logins` `AWSCognitoCredentialsProviderHelper`

```

override func logins () -> AWSTask<NSDictionary> {
    if(/*logic to determine if user is unauthenticated*/) {
        return AWSTask(result:nil)
    }else if (/*logic to determine if user is Facebook*/){
        if let token = AccessToken.current?.authenticationToken {
            return AWSTask(result: [AWSIdentityProviderFacebook:token])
        }
        return AWSTask(error:NSError(domain: "Facebook Login", code: -1 , userInfo:
["Facebook" : "No current Facebook access token"]))
    }else {
        return super.logins()
    }
}
}

```

## JavaScript

Setelah Anda mendapatkan ID identitas dan token sesi dari backend Anda, Anda akan meneruskannya ke penyedia. `AWS.CognitoIdentityCredentials` Inilah contohnya.

```

AWS.config.credentials = new AWS.CognitoIdentityCredentials({
    IdentityPoolId: 'IDENTITY_POOL_ID',
    IdentityId: 'IDENTITY_ID_RETURNED_FROM_YOUR_PROVIDER',
    Logins: {
        'cognito-identity.amazonaws.com': 'TOKEN_RETURNED_FROM_YOUR_PROVIDER'
    }
});

```

## Unity

Untuk menggunakan identitas yang diautentikasi pengembang, Anda harus memperluas `CognitoAWSCredentials` dan mengganti `RefreshIdentity` metode untuk mengambil id identitas pengguna dan token dari backend Anda dan mengembalikannya. Berikut ini adalah contoh sederhana dari penyedia identitas yang akan menghubungi backend hipotetis di 'example.com':

```

using UnityEngine;
using System.Collections;
using Amazon.CognitoIdentity;
using System.Collections.Generic;
using ThirdParty.Json.LitJson;
using System;
using System.Threading;

```

```
public class DeveloperAuthenticatedCredentials : CognitoAWSCredentials
{
    const string PROVIDER_NAME = "example.com";
    const string IDENTITY_POOL = "IDENTITY_POOL_ID";
    static readonly RegionEndpoint REGION = RegionEndpoint.USEast1;

    private string login = null;

    public DeveloperAuthenticatedCredentials(string loginAlias)
        : base(IDENTITY_POOL, REGION)
    {
        login = loginAlias;
    }

    protected override IdentityState RefreshIdentity()
    {
        IdentityState state = null;
        ManualResetEvent waitLock = new ManualResetEvent(false);
        MainThreadDispatcher.ExecuteCoroutineOnMainThread(ContactProvider((s) =>
        {
            state = s;
            waitLock.Set();
        })));
        waitLock.WaitOne();
        return state;
    }

    IEnumerator ContactProvider(Action<IdentityState> callback)
    {
        WWW www = new WWW("http://example.com/?username="+login);
        yield return www;
        string response = www.text;

        JsonData json = JsonMapper.ToObject(response);

        //The backend has to send us back an Identity and a OpenID token
        string identityId = json["IdentityId"].ToString();
        string token = json["Token"].ToString();

        IdentityState state = new IdentityState(identityId, PROVIDER_NAME, token,
false);
        callback(state);
    }
}
```

```
}
```

Kode di atas menggunakan objek operator thread untuk memanggil coroutine. Jika Anda tidak memiliki cara untuk melakukan ini di proyek Anda, Anda dapat menggunakan skrip berikut dalam scene Anda:

```
using System;
using UnityEngine;
using System.Collections;
using System.Collections.Generic;

public class MainThreadDispatcher : MonoBehaviour
{
    static Queue<IEnumerator> _coroutineQueue = new Queue<IEnumerator>();
    static object _lock = new object();

    public void Update()
    {
        while (_coroutineQueue.Count > 0)
        {
            StartCoroutine(_coroutineQueue.Dequeue());
        }
    }

    public static void ExecuteCoroutineOnMainThread(IEnumerator coroutine)
    {
        lock (_lock) {
            _coroutineQueue.Enqueue(coroutine);
        }
    }
}
```

## Xamarin

Untuk menggunakan identitas yang diautentikasi pengembang, Anda harus memperluas `CognitoAWSCredentials` dan mengganti `RefreshIdentity` metode untuk mengambil id identitas pengguna dan token dari backend Anda dan mengembalikannya. Berikut ini adalah contoh dasar dari penyedia identitas yang akan menghubungi backend hipotetis di 'example.com':

```
public class DeveloperAuthenticatedCredentials : CognitoAWSCredentials
{
```

```
const string PROVIDER_NAME = "example.com";
const string IDENTITY_POOL = "IDENTITY_POOL_ID";
static readonly RegionEndpoint REGION = RegionEndpoint.USEast1;
private string login = null;

public DeveloperAuthenticatedCredentials(string loginAlias)
    : base(IDENTITY_POOL, REGION)
{
    login = loginAlias;
}

protected override async Task<IdentityState> RefreshIdentityAsync()
{
    IdentityState state = null;
    //get your identity and set the state
    return state;
}
}
```

## Memperbarui peta login (hanya Android dan iOS)

### Android

Setelah berhasil mengautentikasi pengguna dengan sistem otentikasi Anda, perbarui peta login dengan nama penyedia pengembang dan pengenalan pengguna pengembang. Ini adalah string alfanumerik yang secara unik mengidentifikasi pengguna dalam sistem otentikasi Anda. Pastikan untuk memanggil metode `refresh` setelah memperbarui peta login sebagai `identityId` mungkin telah berubah:

```
HashMap<String, String> loginsMap = new HashMap<String, String>();
loginsMap.put(developerAuthenticationProvider.getProviderName(),
    developerUserIdentifier);

credentialsProvider.setLogins(loginsMap);
credentialsProvider.refresh();
```

### iOS - Objektif-C

SDK iOS hanya memanggil metode `logins` untuk mendapatkan peta login terbaru jika tidak ada kredensial atau kredensial telah kedaluwarsa. Jika Anda ingin memaksa SDK untuk mendapatkan kredensi baru (misalnya, pengguna akhir Anda beralih dari yang tidak diautentikasi ke yang



diautentikasi dan Anda ingin kredensialnya terhadap pengguna yang diautentikasi), hubungi `clearCredentials` di `credentialsProvider`

```
[credentialsProvider clearCredentials];
```

## iOS - cepat

SDK iOS hanya memanggil metode `logins` untuk mendapatkan peta login terbaru jika tidak ada kredensial atau kredensial telah kedaluwarsa. Jika Anda ingin memaksa SDK untuk mendapatkan kredensial baru (misalnya, pengguna akhir Anda beralih dari tidak diautentikasi menjadi diautentikasi dan Anda ingin kredensial terhadap pengguna yang diautentikasi), panggil `clearCredentials` di `credentialsProvider` Anda.

```
credentialsProvider.clearCredentials()
```

## Mendapatkan token (sisi server)

Anda mendapatkan token dengan menelepon [GetOpenIdTokenForDeveloperIdentity](#). API ini harus dipanggil dari backend Anda menggunakan AWS kredensi pengembang. Itu tidak boleh dipanggil dari SDK klien. API menerima ID kumpulan identitas Cognito; peta login yang berisi nama penyedia identitas Anda sebagai kunci dan pengenal sebagai nilai; dan secara opsional ID identitas Cognito (misalnya, Anda membuat pengguna yang tidak diautentikasi diautentikasi). Pengidentifikasi dapat berupa nama pengguna dari pengguna Anda, alamat email, atau nilai numerik. API merespons panggilan Anda dengan ID Cognito unik untuk pengguna Anda dan token OpenID Connect untuk pengguna akhir.

Beberapa hal yang perlu diingat tentang token yang dikembalikan oleh `GetOpenIdTokenForDeveloperIdentity`:

- Anda dapat menentukan waktu kedaluwarsa kustom untuk token sehingga Anda dapat men-cache token itu. Jika Anda tidak memberikan waktu kedaluwarsa kustom, token ini berlaku selama 15 menit.
- Durasi token maksimum yang dapat Anda atur adalah 24 jam.
- Berhati-hati terhadap implikasi keamanan dari meningkatkan durasi token. Jika penyerang memperoleh token ini, mereka dapat menukarnya dengan AWS kredensi untuk pengguna akhir selama durasi token.

Cuplikan Java berikut menunjukkan cara menginisialisasi klien Amazon Cognito dan mengambil token untuk identitas yang diautentikasi pengembang.

```
// authenticate your end user as appropriate
// ....

// if authenticated, initialize a cognito client with your AWS developer credentials
AmazonCognitoIdentity identityClient = new AmazonCognitoIdentityClient(
    new BasicAWSCredentials("access_key_id", "secret_access_key")
);

// create a new request to retrieve the token for your end user
GetOpenIdTokenForDeveloperIdentityRequest request =
    new GetOpenIdTokenForDeveloperIdentityRequest();
request.setIdentityPoolId("YOUR_COGNITO_IDENTITY_POOL_ID");

request.setIdentityId("YOUR_COGNITO_IDENTITY_ID"); //optional, set this if your client
    has an
                                                    //identity ID that you want to link
    to this
                                                    //developer account

// set up your logins map with the username of your end user
HashMap<String,String> logins = new HashMap<>();
logins.put("YOUR_IDENTITY_PROVIDER_NAME", "YOUR_END_USER_IDENTIFIER");
request.setLogins(logins);

// optionally set token duration (in seconds)
request.setTokenDuration(60 * 151);
GetOpenIdTokenForDeveloperIdentityResult response =
    identityClient.getOpenIdTokenForDeveloperIdentity(request);

// obtain identity id and token to return to your client
String identityId = response.getIdentityId();
String token = response.getToken();

//code to return identity id and token to client
//...
```

Mengikuti langkah-langkah sebelumnya, Anda harus dapat mengintegrasikan identitas yang diautentikasi pengembang di aplikasi Anda. Jika Anda memiliki masalah atau pertanyaan, jangan ragu untuk memposting di [forum](#) kami.

## Connect ke identitas sosial yang ada

Semua penautan penyedia ketika Anda menggunakan identitas yang diautentikasi pengembang harus dilakukan dari backend Anda. Untuk menghubungkan identitas kustom ke identitas sosial pengguna (Login with Amazon, Masuk dengan Apple, Facebook, atau Google), tambahkan token penyedia identitas ke peta login saat Anda menelepon [GetOpenIdTokenForDeveloperIdentity](#). Untuk membuat ini mungkin, ketika Anda memanggil backend Anda dari SDK klien Anda untuk mengautentikasi pengguna akhir Anda, berikan secara tambahan token penyedia sosial pengguna akhir.

Misalnya, jika Anda mencoba untuk menautkan identitas kustom ke Facebook, Anda akan menambahkan token Facebook selain pengidentifikasi penyedia identitas Anda ke peta login ketika Anda memanggil `GetOpenIdTokenForDeveloperIdentity`.

```
logins.put("YOUR_IDENTITY_PROVIDER_NAME", "YOUR_END_USER_IDENTIFIER");
logins.put("graph.facebook.com", "END_USERS_FACEBOOK_ACCESSTOKEN");
```

## Mendukung transisi antar penyedia

### Android

Aplikasi Anda mungkin memerlukan dukungan identitas yang tidak diautentikasi atau identitas yang diautentikasi menggunakan penyedia publik (Login with Amazon, Masuk dengan Apple, Facebook, atau Google) bersama dengan identitas yang diautentikasi pengembang. Perbedaan penting antara identitas yang diautentikasi pengembang dan identitas lainnya (identitas yang tidak diautentikasi dan identitas yang diautentikasi menggunakan penyedia publik) adalah cara IdentityID dan token diperoleh. Untuk identitas lain, aplikasi seluler akan berinteraksi langsung dengan Amazon Cognito alih-alih menghubungi sistem otentikasi Anda. Jadi aplikasi seluler harus dapat mendukung dua alur berbeda tergantung pada pilihan yang dibuat oleh pengguna aplikasi. Untuk ini, Anda harus membuat beberapa perubahan pada penyedia identitas khusus.

`refreshMetode` ini memeriksa peta login. Jika peta tidak kosong dan memiliki kunci dengan nama penyedia pengembang, maka panggil backend Anda. Jika tidak, panggil `getIdentityId` metode dan kembalikan null.

```
public String refresh() {

    setToken(null);
```

```
// If the logins map is not empty make a call to your backend
// to get the token and identityId
if (getProviderName() != null &&
    !this.loginsMap.isEmpty() &&
    this.loginsMap.containsKey(getProviderName())) {

    /**
     * This is where you would call your backend
     **/

    // now set the returned identity id and token in the provider
    update(identityId, token);
    return token;

} else {
    // Call getIdentityId method and return null
    this.getIdentityId();
    return null;
}
}
```

Demikian pula metode `getIdentityId` akan memiliki dua alur tergantung pada isi dari peta login:

```
public String getIdentityId() {

    // Load the identityId from the cache
    identityId = cachedIdentityId;

    if (identityId == null) {

        // If the logins map is not empty make a call to your backend
        // to get the token and identityId

        if (getProviderName() != null && !this.loginsMap.isEmpty()
            && this.loginsMap.containsKey(getProviderName())) {

            /**
             * This is where you would call your backend
             **/

            // now set the returned identity id and token in the provider
            update(identityId, token);
            return token;

        }

    }

}
```

```
    } else {
        // Otherwise call &COG; using getIdentityId of super class
        return super.getIdentityId();
    }

} else {
    return identityId;
}

}
```

## iOS - Objektiv-C

Aplikasi Anda mungkin memerlukan dukungan identitas yang tidak diautentikasi atau identitas yang diautentikasi menggunakan penyedia publik (Login with Amazon, Masuk dengan Apple, Facebook, atau Google) bersama dengan identitas yang diautentikasi pengembang. Untuk melakukan ini, ganti [AWSCognitoCredentialsProviderHelper](#) logins metode untuk dapat mengembalikan peta login yang benar berdasarkan penyedia identitas saat ini. Contoh ini menunjukkan kepada Anda bagaimana Anda dapat berputar antara tidak diautentikasi, Facebook, dan yang diautentikasi oleh pengembang.

```
- (AWSTask<NSDictionary<NSString *, NSString *> *)logins {
    if(/*logic to determine if user is unauthenticated*/) {
        return [AWSTask taskWithResult:nil];
    }else if (/*logic to determine if user is Facebook*/) {
        return [AWSTask taskWithResult: @{ AWSIdentityProviderFacebook :
[FBSDKAccessToken currentAccessToken] }];
    }else {
        return [super logins];
    }
}
```

Ketika Anda beralih dari tidak diautentikasi ke diautentikasi, Anda harus memanggil `[credentialsProvider clearCredentials];` untuk memaksa SDK untuk mendapatkan kredensial baru yang diautentikasi. Ketika Anda beralih di antara dua penyedia yang diautentikasi dan Anda tidak mencoba menautkan kedua penyedia (misalnya, Anda tidak menyediakan token untuk beberapa penyedia di kamus login Anda), hubungi `[credentialsProvider clearKeychain];` Ini akan menghapus baik kredensial dan identitas dan memaksa SDK untuk mendapatkan yang baru.

## iOS - cepat

Aplikasi Anda mungkin memerlukan dukungan identitas yang tidak diautentikasi atau identitas yang diautentikasi menggunakan penyedia publik (Login with Amazon, Masuk dengan Apple, Facebook, atau Google) bersama dengan identitas yang diautentikasi pengembang. Untuk melakukan ini, ganti [AWSCognitoCredentialsProviderHelper.logins](#) metode untuk dapat mengembalikan peta login yang benar berdasarkan penyedia identitas saat ini. Contoh ini menunjukkan kepada Anda bagaimana Anda dapat berputar antara tidak diautentikasi, Facebook, dan yang diautentikasi oleh pengembang.

```
override func logins () -> AWSTask<NSDictionary> {
    if(/*logic to determine if user is unauthenticated*/) {
        return AWSTask(result:nil)
    }else if (/*logic to determine if user is Facebook*/) {
        if let token = AccessToken.current?.authenticationToken {
            return AWSTask(result: [AWSIdentityProviderFacebook:token])
        }
        return AWSTask(error: NSError(domain: "Facebook Login", code: -1 , userInfo:
["Facebook" : "No current Facebook access token"]))
    }else {
        return super.logins()
    }
}
```

Ketika Anda beralih dari tidak diautentikasi ke diautentikasi, Anda harus memanggil `credentialsProvider.clearCredentials()` untuk memaksa SDK untuk mendapatkan kredensial baru yang diautentikasi. Ketika Anda beralih di antara dua penyedia yang diautentikasi dan Anda tidak mencoba menautkan kedua penyedia tersebut (yaitu Anda tidak menyediakan token untuk beberapa penyedia dalam kamus login Anda), Anda harus memanggil `credentialsProvider.clearKeychain()`. Ini akan menghapus baik kredensial dan identitas dan memaksa SDK untuk mendapatkan yang baru.

## Unity

Aplikasi Anda mungkin memerlukan dukungan identitas yang tidak diautentikasi atau identitas yang diautentikasi menggunakan penyedia publik (Login with Amazon, Masuk dengan Apple, Facebook, atau Google) bersama dengan identitas yang diautentikasi pengembang. Perbedaan penting antara identitas yang diautentikasi pengembang dan identitas lainnya (identitas yang tidak diautentikasi dan identitas yang diautentikasi menggunakan penyedia publik) adalah cara IdentityID dan token diperoleh. Untuk identitas lain, aplikasi seluler akan berinteraksi langsung dengan Amazon Cognito alih-alih menghubungi sistem otentikasi Anda. Aplikasi seluler harus dapat mendukung dua aliran

berbeda tergantung pada pilihan yang dibuat oleh pengguna aplikasi. Untuk ini, Anda harus membuat beberapa perubahan pada penyedia identitas kustom.

Cara yang disarankan untuk melakukannya di Unity adalah dengan memperluas penyedia identitas Anda dari `AmazonCognitoEnhancedIdentityProvider` alih-alih `AbstractCognitoIdentityProvider`, dan memanggil `RefreshAsync` metode induk alih-alih metode Anda sendiri jika pengguna tidak diautentikasi dengan backend Anda sendiri. Jika pengguna diautentikasi, Anda dapat menggunakan alur yang sama yang dijelaskan sebelumnya.

## Xamarin

Aplikasi Anda mungkin memerlukan dukungan identitas yang tidak diautentikasi atau identitas yang diautentikasi menggunakan penyedia publik (Login with Amazon, Masuk dengan Apple, Facebook, atau Google) bersama dengan identitas yang diautentikasi pengembang. Perbedaan penting antara identitas yang diautentikasi pengembang dan identitas lainnya (identitas yang tidak diautentikasi dan identitas yang diautentikasi menggunakan penyedia publik) adalah cara `IdentityID` dan token diperoleh. Untuk identitas lain, aplikasi seluler akan berinteraksi langsung dengan Amazon Cognito alih-alih menghubungi sistem otentikasi Anda. Aplikasi seluler harus dapat mendukung dua aliran berbeda tergantung pada pilihan yang dibuat oleh pengguna aplikasi. Untuk ini, Anda harus membuat beberapa perubahan pada penyedia identitas khusus.

## Mengalihkan pengguna yang tidak diautentikasi ke pengguna terautentikasi (kumpulan identitas)

Kolam identitas Amazon Cognito mendukung pengguna yang diautentikasi dan tidak terautentikasi. Pengguna yang tidak diautentikasi menerima akses ke AWS sumber daya Anda bahkan jika mereka tidak masuk dengan penyedia identitas (IdPs) mana pun. Tingkat akses ini berguna untuk menampilkan konten kepada pengguna sebelum mereka masuk. Setiap pengguna tidak terautentikasi memiliki identitas unik di kolam identitas, meskipun mereka belum masuk secara individual dan diautentikasi.

Bagian ini menjelaskan kasus di mana pengguna Anda memilih untuk beralih dari masuk dengan identitas yang tidak terautentikasi ke menggunakan identitas yang terautentikasi.

## Android

Pengguna dapat masuk ke aplikasi Anda sebagai tamu yang tidak terautentikasi. Akhirnya mereka mungkin memutuskan untuk masuk menggunakan salah satu yang didukung IdPs. Amazon Cognito

memastikan bahwa identitas lama mempertahankan pengenal unik yang sama seperti yang baru, dan bahwa data profil digabung secara otomatis.

Aplikasi Anda diberitahu tentang gabungan profil melalui antarmuka `IdentityChangedListener`. Menerapkan metode `identityChanged` dalam antarmuka untuk menerima pesan ini:

```
@override
public void identityChanged(String oldIdentityId, String newIdentityId) {
    // handle the change
}
```

## iOS - Objective-C

Pengguna dapat masuk ke aplikasi Anda sebagai tamu yang tidak terautentikasi. Akhirnya mereka mungkin memutuskan untuk masuk menggunakan salah satu yang didukung IdPs. Amazon Cognito memastikan bahwa identitas lama mempertahankan pengenal unik yang sama seperti yang baru, dan bahwa data profil digabung secara otomatis.

`NSNotificationCenter` menginformasikan aplikasi penggabungan profil Anda:

```
[[NSNotificationCenter defaultCenter] addObserver:self
                                       selector:@selector(identityIdDidChange:)
                                       name:AWSCognitoIdentityIdChangedNotification
                                       object:nil];

-(void)identityDidChange:(NSNotification*)notification {
    NSDictionary *userInfo = notification.userInfo;
    NSLog(@"identity changed from %@ to %@",
          [userInfo objectForKey:AWSCognitoNotificationPreviousId],
          [userInfo objectForKey:AWSCognitoNotificationNewId]);
}
```

## iOS - Swift

Pengguna dapat masuk ke aplikasi Anda sebagai tamu yang tidak terautentikasi. Akhirnya mereka mungkin memutuskan untuk masuk menggunakan salah satu yang didukung IdPs. Amazon Cognito memastikan bahwa identitas lama mempertahankan pengenal unik yang sama seperti yang baru, dan bahwa data profil digabung secara otomatis.

`NSNotificationCenter` menginformasikan aplikasi penggabungan profil Anda:



```
[NSNotificationCenter defaultCenter().addObserver(observer: self
 selector:"identityDidChange"
 name:AWSCognitoIdentityIdChangedNotification
 object:nil)

func identityDidChange(notification: NSNotification!) {
    if let userInfo = notification.userInfo as? [String: AnyObject] {
        print("identity changed from: \(userInfo[AWSCognitoNotificationPreviousId])
 to: \(userInfo[AWSCognitoNotificationNewId])")
    }
}
```

## JavaScript

### Pengguna Awalnya Tidak Diautentikasi

Pengguna biasanya mulai dengan peran tidak terautentikasi. Untuk peran ini, Anda mengatur properti kredensial dari objek konfigurasi Anda tanpa properti Masuk. Dalam kasus ini, konfigurasi default Anda mungkin terlihat seperti berikut ini:

```
// set the default config object
var creds = new AWS.CognitoIdentityCredentials({
    IdentityPoolId: 'us-east-1:1699ebc0-7900-4099-b910-2df94f52a030'
});
AWS.config.credentials = creds;
```

### Beralih ke Pengguna Terautentikasi

Ketika pengguna tidak terautentikasi masuk ke IdP dan Anda memiliki token, Anda dapat beralih pengguna dari yang tidak terautentikasi menjadi terautentikasi dengan memanggil fungsi kustom yang memperbarui kredensial objek dan menambahkan token Masuk:

```
// Called when an identity provider has a token for a logged in user
function userLoggedIn(providerName, token) {
    creds.params.Logins = creds.params.Logins || {};
    creds.params.Logins[providerName] = token;

    // Expire credentials to refresh them on the next request
    creds.expired = true;
```

```
}
```

Anda juga dapat membuat objek `CognitoIdentityCredentials`. Jika Anda melakukannya, Anda harus mengatur ulang properti kredensial objek layanan yang sudah ada untuk mencerminkan informasi konfigurasi kredensial yang diperbarui. Lihat [Menggunakan objek konfigurasi global](#).

Untuk informasi selengkapnya tentang `CognitoIdentityCredentials` objek, lihat [AWS CognitoIdentityCredentials](#) dalam Referensi AWS SDK for JavaScript API.

## Unity

Pengguna dapat masuk ke aplikasi Anda sebagai tamu yang tidak terautentikasi. Akhirnya mereka mungkin memutuskan untuk masuk menggunakan salah satu yang didukung IdPs. Amazon Cognito memastikan bahwa identitas lama mempertahankan pengenal unik yang sama seperti yang baru, dan bahwa data profil digabung secara otomatis.

Anda dapat berlangganan ke `IdentityChangedEvent` untuk menerima notifikasi tentang penggabungan profil:

```
credentialsProvider.IdentityChangedEvent += delegate(object sender,
    CognitoAWSCredentials.IdentityChangedEventArgs e)
{
    // handle the change
    Debug.Log("Identity changed from " + e.OldIdentityId + " to " + e.NewIdentityId);
};
```

## Xamarin

Pengguna dapat masuk ke aplikasi Anda sebagai tamu yang tidak terautentikasi. Akhirnya mereka mungkin memutuskan untuk masuk menggunakan salah satu yang didukung IdPs. Amazon Cognito memastikan bahwa identitas lama mempertahankan pengenal unik yang sama seperti yang baru, dan bahwa data profil digabung secara otomatis.

```
credentialsProvider.IdentityChangedEvent += delegate(object sender,
    CognitoAWSCredentials.IdentityChangedEventArgs e){
    // handle the change
    Console.WriteLine("Identity changed from " + e.OldIdentityId + " to " +
    e.NewIdentityId);
};
```

# Amazon Cognito Sync

**⚠** Jika Anda baru mengenal Amazon Cognito Sync, gunakan [AWS AppSync](#). Seperti Amazon Cognito Sync, AWS AppSync adalah layanan untuk menyinkronkan data aplikasi di seluruh perangkat.  
Hal ini memungkinkan data pengguna seperti preferensi aplikasi atau game state agar dapat disinkronkan. Hal ini juga memperluas kemampuan ini dengan memungkinkan beberapa pengguna untuk menyinkronkan dan berkolaborasi secara langsung pada data bersama.

Amazon Cognito Sync adalah pustaka klien Layanan AWS dan yang memungkinkan untuk menyinkronkan data pengguna terkait aplikasi di seluruh perangkat. Amazon Cognito Sync dapat menyinkronkan data profil pengguna di seluruh perangkat seluler dan web tanpa menggunakan backend Anda sendiri. Library klien menyimpan data cache secara lokal sehingga aplikasi Anda dapat membaca dan menulis data terlepas dari status konektivitas perangkat. Saat perangkat online, Anda dapat menyinkronkan data. Jika Anda mengatur sinkronisasi push, Anda dapat segera memberi tahu perangkat lain bahwa pembaruan tersedia.

Untuk informasi tentang ketersediaan Wilayah identitas Amazon Cognito, lihat [AWS Ketersediaan Wilayah Layanan](#).

Untuk mempelajari selengkapnya tentang Amazon Cognito Sync, lihat topik berikut.

## Topik

- [Memulai dengan Amazon Cognito Sync](#)
- [Menyinkronkan data](#)
- [Penanganan callback](#)
- [Sinkronisasi push](#)
- [Pengaliran Amazon Cognito](#)
- [Peristiwa Amazon Cognito](#)

# Memulai dengan Amazon Cognito Sync

**⚠** Jika Anda baru mengenal Amazon Cognito Sync, gunakan [AWS AppSync](#). Seperti Amazon Cognito Sync, AWS AppSync adalah layanan untuk menyinkronkan data aplikasi di seluruh perangkat.

Hal ini memungkinkan data pengguna seperti preferensi aplikasi atau game state agar dapat disinkronkan. Hal ini juga memperluas kemampuan ini dengan memungkinkan beberapa pengguna untuk menyinkronkan dan berkolaborasi secara langsung pada data bersama.

Amazon Cognito Sync adalah sebuah layanan AWS dan perpustakaan klien yang memungkinkan sinkronisasi lintas-perangkat terhadap data pengguna terkait aplikasi. Anda dapat menggunakannya untuk menyinkronkan data profil pengguna di perangkat seluler dan aplikasi web. Perpustakaan klien melakukan cache data secara lokal sehingga aplikasi Anda dapat membaca dan menulis data terlepas dari status konektivitas perangkat. Saat perangkat sedang online, Anda dapat menyinkronkan data, dan jika Anda mengatur sinkronisasi push, segera beri tahu perangkat lain bahwa pembaruan tersedia.

## Menyiapkan kumpulan identitas di Amazon Cognito

Amazon Cognito Sync membutuhkan kolam identitas Amazon Cognito untuk memberikan identitas pengguna. Sebelum menggunakan Amazon Cognito Sync, Anda harus terlebih dahulu menyiapkan kumpulan identitas. Untuk membuat pangkalan identitas dan menginstal SDK, lihat [Memulai dengan kumpulan identitas Amazon Cognito](#).

## Menyimpan dan menyinkronkan data

Setelah menyiapkan pangkalan identitas dan menginstal SDK, Anda dapat mulai menyimpan dan menyinkronkan data antar perangkat. Untuk informasi selengkapnya, lihat [Menyinkronkan data](#).

## Menyinkronkan data

**⚠** Jika Anda baru mengenal Amazon Cognito Sync, gunakan [AWS AppSync](#). Seperti Amazon Cognito Sync, AWS AppSync adalah layanan untuk menyinkronkan data aplikasi di seluruh perangkat.

Hal ini memungkinkan data pengguna seperti preferensi aplikasi atau game state agar dapat disinkronkan. Hal ini juga memperluas kemampuan ini dengan memungkinkan beberapa pengguna untuk menyinkronkan dan berkolaborasi secara langsung pada data bersama.

Dengan Amazon Cognito, Anda dapat menyimpan data pengguna dalam kumpulan data yang berisi pasangan nilai-kunci. Amazon Cognito mengaitkan data ini dengan identitas di kumpulan identitas Anda sehingga aplikasi Anda dapat mengaksesnya di seluruh login dan perangkat. Untuk menyinkronkan data ini di antara layanan Amazon Cognito dan perangkat pengguna akhir, gunakan metode sinkronisasi. Setiap set data dapat memiliki ukuran maksimum 1 MB. Anda dapat mengaitkan hingga 20 set data dengan identitas.

Amazon Cognito Sync client membuat cache lokal untuk data identitas. Saat aplikasi membaca dan menulis kunci, aplikasi akan berkomunikasi dengan cache lokal ini. Komunikasi ini menjamin bahwa semua perubahan yang Anda lakukan pada perangkat segera tersedia di perangkat, bahkan saat Anda sedang offline. Ketika metode sinkronisasi dipanggil, perubahan dari layanan ditarik ke perangkat, dan setiap perubahan lokal didorong ke layanan. Pada titik ini, perubahan tersedia untuk perangkat lain untuk disinkronkan.

## Menginisialisasi klien Amazon Cognito Sync

Untuk menginisialisasi klien Amazon Cognito Sync, Anda harus terlebih dahulu membuat penyedia kredensial. Penyedia kredensial memperoleh AWS kredensial sementara untuk memungkinkan aplikasi Anda mengakses sumber daya Anda. AWS Anda juga harus mengimpor file header yang diperlukan. Gunakan langkah-langkah berikut untuk menginisialisasi Amazon Cognito Sync client.

### Android

1. Buat penyedia kredensial, ikuti petunjuk di [Mendapatkan kredensial](#).
2. Impor paket Amazon Cognito sebagai berikut: `import com.amazonaws.mobileconnectors.cognito.*;`
3. Inisialisasi Sinkronisasi Amazon Cognito. Teruskan konteks aplikasi Android, ID pangkalan identitas Wilayah AWS, dan penyedia kredensial Amazon Cognito yang diinisialisasi sebagai berikut:

```
CognitoSyncManager client = new CognitoSyncManager(  
    getApplicationContext(),  
    Regions.YOUR_REGION,
```

```
credentialsProvider);
```

## iOS - Objective-C

1. Buat penyedia kredensial, ikuti petunjuk di [Mendapatkan kredensial](#).
2. Impor AWSCore danCognito, dan inialisasi AWSCognito sebagai berikut:

```
#import <AWSiOSSDKv2/AWSCore.h>
#import <AWSCognitoSync/Cognito.h>

AWSCognito *syncClient = [AWSCognito defaultCognito];
```

3. Jika Anda menggunakanCocoaPods, ganti <AWSiOSSDKv2/AWSCore.h> denganAWSCore.h. Ikuti sintaks yang sama untuk impor Amazon Cognito.

## iOS - Swift

1. Buat penyedia kredensial, ikuti petunjuk di [Mendapatkan kredensial](#).
2. Impor dan inialisasi AWSCognito sebagai berikut:

```
import AWSCognito
let syncClient = AWSCognito.default()!
```

## JavaScript

1. Unduh [Amazon Cognito Sync Manager untuk JavaScript](#).
2. Sertakan perpustakaan Sync Manager di proyek Anda.
3. Buat penyedia kredensial, ikuti petunjuk di [Mendapatkan kredensial](#).
4. Inialisasi Manajer Sinkronisasi sebagai berikut:

```
var syncManager = new AWS.CognitoSyncManager();
```

## Unity

1. Buat sebuah instance dari `CognitoAWSCredentials`, mengikuti petunjuk di [Mendapatkan kredensial](#).
2. Buat instans `CognitoSyncManager`. Lulus `CognitoAwsCredentials` objek dan `AmazonCognitoSyncConfig`, dan sertakan setidaknya `Region` set, sebagai berikut:

```
AmazonCognitoSyncConfig clientConfig = new AmazonCognitoSyncConfig { RegionEndpoint =  
    REGION };  
CognitoSyncManager syncManager = new CognitoSyncManager(credentials, clientConfig);
```

## Xamarin

1. Buat sebuah instance dari `CognitoAWSCredentials`, mengikuti petunjuk di [Mendapatkan kredensial](#).
2. Buat instans `CognitoSyncManager`. Lulus `CognitoAwsCredentials` objek dan `AmazonCognitoSyncConfig`, dan sertakan setidaknya `Region` set, sebagai berikut:

```
AmazonCognitoSyncConfig clientConfig = new AmazonCognitoSyncConfig { RegionEndpoint =  
    REGION };  
CognitoSyncManager syncManager = new CognitoSyncManager(credentials, clientConfig);
```

## Memahami kumpulan data

Amazon Cognito mengatur data profil pengguna ke dalam kumpulan data. Setiap set data dapat berisi hingga 1MB data dalam bentuk pasangan nilai kunci. Dataset adalah entitas paling terperinci yang dapat Anda sinkronkan. Membaca dan menulis operasi yang dilakukan pada set data hanya mempengaruhi toko lokal sampai metode sinkronisasi dipanggil. Amazon Cognito mengidentifikasi kumpulan data dengan string unik. Anda dapat membuat dataset baru atau membuka yang sudah ada sebagai berikut.

## Android

```
Dataset dataset = client.openOrCreateDataset("datasetname");
```

Untuk menghapus kumpulan data, pertama-tama panggil metode untuk menghapusnya dari penyimpanan lokal, lalu panggil `synchronize` metode untuk menghapus kumpulan data dari Amazon Cognito sebagai berikut:

```
dataset.delete();
dataset.synchronize(syncCallback);
```

## iOS - Objective-C

```
AWSCognitoDataset *dataset = [syncClient openOrCreateDataset:@"myDataSet"];
```

Untuk menghapus kumpulan data, pertama-tama panggil metode untuk menghapusnya dari penyimpanan lokal, lalu panggil `synchronize` metode untuk menghapus kumpulan data dari Amazon Cognito sebagai berikut:

```
[dataset clear];
[dataset synchronize];
```

## iOS - Swift

```
let dataset = syncClient.openOrCreateDataset("myDataSet")!
```

Untuk menghapus kumpulan data, pertama-tama panggil metode untuk menghapusnya dari penyimpanan lokal, lalu panggil `synchronize` metode sebagai berikut: untuk menghapus kumpulan data dari Amazon Cognito:

```
dataset.clear()
dataset.synchronize()
```

## JavaScript

```
syncManager.openOrCreateDataset('myDatasetName', function(err, dataset) {
  // ...
});
```

## Unity

```
string myValue = dataset.Get("myKey");
```



```
dataset.Put("myKey", "newValue");
```

Untuk menghapus kunci dari dataset, gunakan Remove sebagai berikut:

```
dataset.Remove("myKey");
```

## Xamarin

```
Dataset dataset = syncManager.OpenOrCreateDataset("myDatasetName");
```

Untuk menghapus kumpulan data, pertama-tama panggil metode untuk menghapusnya dari penyimpanan lokal, lalu panggil `synchronize` metode untuk menghapus kumpulan data dari Amazon Cognito sebagai berikut:

```
dataset.Delete();  
dataset.SynchronizeAsync();
```

## Membaca dan menulis data dalam kumpulan data

Set data Amazon Cognito berfungsi sebagai kamus, dengan nilai yang dapat diakses oleh kunci. Anda dapat membaca, menambah, atau memodifikasi kunci dan nilai-nilai dari dataset sama seperti jika dataset adalah kamus, seperti yang ditunjukkan dalam contoh berikut.

Perhatikan bahwa nilai yang Anda tulis ke kumpulan data hanya memengaruhi salinan data yang di-cache lokal hingga Anda memanggil metode sinkronisasi.

## Android

```
String value = dataset.get("myKey");  
dataset.put("myKey", "my value");
```

## iOS - Objective-C

```
[dataset setString:@"my value" forKey:@"myKey"];  
NSString *value = [dataset stringForKey:@"myKey"];
```

## iOS - Swift

```
dataset.setString("my value", forKey:"myKey")
```

```
let value = dataset.stringForKey("myKey")
```

## JavaScript

```
dataset.get('myKey', function(err, value) {
  console.log('myRecord: ' + value);
});

dataset.put('newKey', 'newValue', function(err, record) {
  console.log(record);
});

dataset.remove('oldKey', function(err, record) {
  console.log(success);
});
```

## Unity

```
string myValue = dataset.Get("myKey");
dataset.Put("myKey", "newValue");
```

## Xamarin

```
//obtain a value
string myValue = dataset.Get("myKey");

// Create a record in a dataset and synchronize with the server
dataset.OnSyncSuccess += SyncSuccessCallback;
dataset.Put("myKey", "myValue");
dataset.SynchronizeAsync();

void SyncSuccessCallback(object sender, SyncSuccessEventArgs e) {
  // Your handler code here
}
```

## Android

Untuk menghapus kunci dari dataset, gunakan `remove` metode sebagai berikut:

```
dataset.remove("myKey");
```

## iOS - Objective-C

Untuk menghapus kunci dari dataset, gunakan `removeObjectForKey` sebagai berikut:

```
[dataset removeObjectForKey:@"myKey"];
```

## iOS - Swift

Untuk menghapus kunci dari dataset, gunakan `removeObjectForKey` sebagai berikut:

```
dataset.removeObjectForKey("myKey")
```

## Unity

Untuk menghapus kunci dari dataset, gunakan `Remove` sebagai berikut:

```
dataset.Remove("myKey");
```

## Xamarin

Anda dapat menggunakan `Remove` untuk menghapus kunci dari set data:

```
dataset.Remove("myKey");
```

## Menyinkronkan data lokal dengan toko sinkronisasi

### Android

Metode `synchronize` membandingkan data yang di-cache-kan secara lokal dengan data yang disimpan di Amazon Cognito Sync store. Perubahan jarak jauh diambil dari Amazon Cognito Sync store; resolusi konflik dipanggil jika ada konflik yang terjadi; dan nilai yang diperbarui pada perangkat didorong ke layanan. Untuk menyinkronkan set data, panggil metode `synchronize`:

```
dataset.synchronize(syncCallback);
```

Metode `synchronize` menerima implementasi dari antarmuka `SyncCallback`, yang dibahas di bawah ini.

Metode `synchronizeOnConnectivity()` mencoba untuk menyinkronkan ketika konektivitas tersedia. Jika konektivitas segera tersedia, `synchronizeOnConnectivity()` berperilaku seperti `synchronize()`. Jika tidak, metode ini memantau perubahan konektivitas dan melakukan sinkronisasi setelah konektivitas tersedia. Jika `synchronizeOnConnectivity()` dipanggil beberapa kali, hanya permintaan sinkronisasi terakhir yang disimpan, dan hanya panggilan balik terakhir yang akan dipicu. Jika set data atau callback adalah kumpulan sampah, metode ini tidak akan melakukan sinkronisasi, dan callback tidak akan dipicu.

Untuk belajar lebih lanjut tentang sinkronisasi set data dan callback yang berbeda, lihat [Penanganan callback](#).

## iOS - Objective-C

Metode `synchronize` membandingkan data yang di-cache-kan secara lokal dengan data yang disimpan di Amazon Cognito Sync store. Perubahan jarak jauh diambil dari Amazon Cognito Sync store; resolusi konflik dipanggil jika ada konflik yang terjadi; dan nilai yang diperbarui pada perangkat didorong ke layanan. Untuk menyinkronkan set data, panggil metode `synchronize`:

Metode `synchronize` bersifat asinkron dan menampilkan objek `AWSTask` untuk menangani respons:

```
[[dataset synchronize] continueWithBlock:^id(AWSTask *task) {
    if (task.isCancelled) {
        // Task cancelled.
    } else if (task.error) {
        // Error while executing task.
    } else {
        // Task succeeded. The data was saved in the sync store.
    }
    return nil;
}];
```

Metode `synchronizeOnConnectivity` mencoba untuk melakukan sinkronisasi ketika perangkat memiliki konektivitas. Pertama, `synchronizeOnConnectivity` memeriksa konektivitas dan, jika perangkat online, segera memanggil sinkronisasi dan menampilkan objek `AWSTask` yang terkait dengan upaya tersebut.

Jika perangkat sedang offline, `synchronizeOnConnectivity` 1) menjadwalkan sinkronisasi pada saat perangkat online kembali dan 2) mengembalikan `AWSTask` dengan hasil nihil. Sinkronisasi terjadwal hanya berlaku untuk siklus hidup pada objek set data. Data tidak akan disinkronkan jika aplikasi keluar sebelum konektivitas kembali. Jika Anda ingin diberi tahu saat peristiwa terjadi selama sinkronisasi terjadwal, Anda harus menambahkan pengamat notifikasi yang ditemukan di `AWSCognito`.

Untuk belajar lebih lanjut tentang sinkronisasi set data dan callback yang berbeda, lihat [Penanganan callback](#).

## iOS - Swift

Metode `synchronize` membandingkan data yang di-cache-kan secara lokal dengan data yang disimpan di Amazon Cognito Sync store. Perubahan jarak jauh diambil dari Amazon Cognito Sync store; resolusi konflik dipanggil jika ada konflik yang terjadi; dan nilai yang diperbarui pada perangkat didorong ke layanan. Untuk menyinkronkan set data, panggil metode `synchronize`:

Metode `synchronize` bersifat asinkron dan menampilkan objek `AWSTask` untuk menangani respons:

```
dataset.synchronize().continueWith(block: { (task) -> AnyObject? in

    if task.isCancelled {
        // Task cancelled.
    } else if task.error != nil {
        // Error while executing task
    } else {
        // Task succeeded. The data was saved in the sync store.
    }
    return task
})
```

Metode `synchronizeOnConnectivity` mencoba untuk melakukan sinkronisasi ketika perangkat memiliki konektivitas. Pertama, `synchronizeOnConnectivity` memeriksa konektivitas dan, jika perangkat online, segera memanggil `synchronize` dan menampilkan objek `AWSTask` yang terkait dengan upaya tersebut.

Jika perangkat sedang offline, `synchronizeOnConnectivity` 1) menjadwalkan sinkronisasi pada saat perangkat online kembali dan 2) mengembalikan objek `AWSTask` dengan hasil nihil. Sinkronisasi terjadwal hanya berlaku untuk siklus hidup pada objek set data. Data tidak akan disinkronkan jika

aplikasi keluar sebelum konektivitas kembali. Jika Anda ingin diberi tahu saat peristiwa terjadi selama sinkronisasi terjadwal, Anda harus menambahkan pengamat notifikasi yang ditemukan di AWS Cognito.

Untuk belajar lebih lanjut tentang sinkronisasi set data dan callback yang berbeda, lihat [Penanganan callback](#).

## JavaScript

Metode `synchronize` membandingkan data yang di-cache-kan secara lokal dengan data yang disimpan di Amazon Cognito Sync store. Perubahan jarak jauh diambil dari Amazon Cognito Sync store; resolusi konflik dipanggil jika ada konflik yang terjadi; dan nilai yang diperbarui pada perangkat didorong ke layanan. Untuk menyinkronkan set data, panggil metode `synchronize`:

```
dataset.synchronize();
```

Untuk belajar lebih lanjut tentang sinkronisasi set data dan callback yang berbeda, lihat [Penanganan callback](#).

## Unity

Metode sinkronisasi membandingkan data yang di-cache-kan secara lokal dengan data yang disimpan di Amazon Cognito Sync store. Perubahan jarak jauh diambil dari Amazon Cognito Sync store; resolusi konflik dipanggil jika ada konflik yang terjadi; dan nilai yang diperbarui pada perangkat didorong ke layanan. Untuk menyinkronkan set data, panggil metode `synchronize`:

```
dataset.Synchronize();
```

Sinkronisasi akan berjalan secara asinkron dan akan berakhir dengan memanggil salah satu dari beberapa callback Anda dapat tentukan dalam Set data.

Untuk belajar lebih lanjut tentang sinkronisasi set data dan callback yang berbeda, lihat [Penanganan callback](#).

## Xamarin

Metode `synchronize` membandingkan data yang di-cache-kan secara lokal dengan data yang disimpan di Amazon Cognito Sync store. Perubahan jarak jauh diambil dari Amazon Cognito Sync

store; resolusi konflik dipanggil jika ada konflik yang terjadi; dan nilai yang diperbarui pada perangkat didorong ke layanan. Untuk menyinkronkan set data, panggil metode `synchronize`:

```
dataset.SynchronizeAsync();
```

Untuk belajar lebih lanjut tentang sinkronisasi set data dan callback yang berbeda, lihat [Penanganan callback](#).

## Penanganan callback

**⚠** Jika Anda baru mengenal Amazon Cognito Sync, gunakan [AWS AppSync](#). Seperti Amazon Cognito Sync, AWS AppSync adalah layanan untuk menyinkronkan data aplikasi di seluruh perangkat. Hal ini memungkinkan data pengguna seperti preferensi aplikasi atau game state agar dapat disinkronkan. Hal ini juga memperluas kemampuan ini dengan memungkinkan beberapa pengguna untuk menyinkronkan dan berkolaborasi secara langsung pada data bersama.

Bagian ini menjelaskan cara menangani callback.

## Android

### SyncCallbackAntarmuka

Dengan menerapkan antarmuka `SyncCallback`, Anda dapat menerima pemberitahuan di aplikasi Anda tentang sinkronisasi set data. Aplikasi Anda kemudian dapat membuat keputusan aktif tentang menghapus data lokal, menggabungkan profil yang tidak diautentikasi dan terautentikasi, dan menyelesaikan konflik sinkronisasi. Anda harus menerapkan metode berikut, yang diperlukan oleh antarmuka:

- `onSuccess()`
- `onFailure()`
- `onConflict()`
- `onDatasetDeleted()`
- `onDatasetsMerged()`

Perhatikan bahwa, jika Anda tidak ingin menentukan semua callback, Anda juga dapat menggunakan kelas `DefaultSyncCallback` yang menyediakan implementasi kosong dan default untuk mereka semua.

### OnSuccess

Callback `onSuccess()` dipicu saat set data berhasil diunduh dari sync store.

```
@Override
public void onSuccess(Dataset dataset, List<Record> newRecords) {
}
```

### padaKegagalan

`onFailure()` dipanggil jika pengecualian terjadi selama sinkronisasi.

```
@Override
public void onFailure(DataStorageException dse) {
}
```

### padaKonflik

Konflik mungkin timbul jika kunci yang sama telah diubah di local store dan sync store. Metode `onConflict()` menangani resolusi konflik. Jika Anda tidak menerapkan metode ini, Amazon Cognito Sync client menerapkan default dengan menggunakan perubahan terbaru.

```
@Override
public boolean onConflict(Dataset dataset, final List<SyncConflict> conflicts) {
    List<Record> resolvedRecords = new ArrayList<Record>();
    for (SyncConflict conflict : conflicts) {
        /* resolved by taking remote records */
        resolvedRecords.add(conflict.resolveWithRemoteRecord());

        /* alternately take the local records */
        // resolvedRecords.add(conflict.resolveWithLocalRecord());

        /* or customer logic, say concatenate strings */
        // String newValue = conflict.getRemoteRecord().getValue()
        //     + conflict.getLocalRecord().getValue();
        // resolvedRecords.add(conflict.resolveWithValue(newValue);
    }
    dataset.resolve(resolvedRecords);
}
```



```
// return true so that synchronize() is retried after conflicts are resolved
return true;
}
```

## onDatasetDeleted

Ketika set data dihapus, Amazon Cognito client menggunakan antarmuka SyncCallback untuk memastikan apakah salinan yang di-cache-kan secara lokal pada set data harus dihapus juga. Menerapkan metode `onDatasetDeleted()` untuk memberi tahu SDK klien apa yang harus dilakukan dengan data lokal.

```
@Override
public boolean onDatasetDeleted(Dataset dataset, String datasetName) {
    // return true to delete the local copy of the dataset
    return true;
}
```

## onDatasetMerged

Ketika dua identitas yang sebelumnya tidak terhubung dihubungkan bersama-sama, semua set data mereka digabung. Aplikasi akan menerima pemberitahuan tentang penggabungan melalui metode `onDatasetsMerged()`:

```
@Override
public boolean onDatasetsMerged(Dataset dataset, List<String> datasetNames) {
    // return false to handle Dataset merge outside the synchronization callback
    return false;
}
```

## iOS - Objective-C

### Notifikasi Sinkronisasi

Amazon Cognito client akan memancarkan sejumlah peristiwa `NSNotification` selama panggilan sinkronisasi. Anda dapat mendaftar untuk memantau notifikasi ini melalui standar `NSNotificationCenter`:

```
[NSNotificationCenter defaultCenter]
addObserver:self
selector:@selector(myNotificationHandler:)
```

```
name:NOTIFICATION_TYPE  
object:nil];
```

Amazon Cognito mendukung lima jenis notifikasi, sebagaimana yang tercantum di bawah ini.

#### AWSCognitoDidStartSynchronizeNotification

Dipanggil ketika operasi sinkronisasi dimulai. `userInfo` akan berisi set data kunci yang merupakan nama dari set data yang disinkronkan.

#### AWSCognitoDidEndSynchronizeNotification

Disebut ketika operasi sinkronisasi selesai (berhasil atau sebaliknya). `userInfo` akan berisi set data kunci yang merupakan nama dari set data yang disinkronkan.

#### AWSCognitoDidFailToSynchronizeNotification

Dipanggil ketika operasi sinkronisasi gagal. `userInfo` akan berisi set data kunci yang merupakan nama set data yang disinkronkan dan kesalahan kunci yang akan berisi kesalahan yang menyebabkan kegagalan.

#### AWSCognitoDidChangeRemoteValueNotification

Dipanggil ketika perubahan lokal berhasil didorong ke Amazon Cognito. `userInfo` akan berisi set data kunci yang merupakan nama set data yang disinkronkan dan kunci kunci yang akan berisi NSArray kunci record yang didorong.

#### AWSCognitoDidChangeLocalValueFromRemoteNotification

Dipanggil ketika nilai lokal berubah akibat operasi sinkronisasi. `userInfo` akan berisi set data kunci yang merupakan nama set data yang disinkronkan dan kunci kunci yang akan berisi NSArray kunci record yang telah berubah.

### Penangan Resolusi Konflik

Selama operasi sinkronisasi, konflik mungkin timbul jika kunci yang sama telah diubah di local store dan sync store. Jika Anda belum menetapkan penanganan resolusi konflik, Amazon Cognito menetapkan setelan default yaitu memilih pembaruan terbaru.

Dengan menerapkan dan menetapkan `AWSCognitoRecordConflictHandler` Anda dapat mengubah resolusi konflik default. Konflik parameter input `AWSCognitoConflict` berisi objek `AWSCognitoRecord` untuk data yang di-cache-kan secara lokal dan catatan yang bertentangan di sync store.

Menggunakan `AWSCognitoConflict` Anda dapat menyelesaikan konflik dengan catatan lokal: [konflik `resolveWithLocal` Rekam], catatan jarak jauh: [konflik `resolveWithRemote` Rekam] atau nilai baru: [resolveWithValueconflict:value]. Menampilkan nil dari metode ini mencegah sinkronisasi berlanjut dan konflik akan ditampilkan lagi saat proses sinkronisasi dimulai kembali.

Anda dapat mengatur penanganan resolusi konflik di tingkat klien:

```
client.conflictHandler = ^AWSCognitoResolvedConflict* (NSString *datasetName,
    AWSCognitoConflict *conflict) {
    // always choose local changes
    return [conflict resolveWithLocalRecord];
};
```

Atau pada tingkat set data:

```
dataset.conflictHandler = ^AWSCognitoResolvedConflict* (NSString *datasetName,
    AWSCognitoConflict *conflict) {
    // override and always choose remote changes
    return [conflict resolveWithRemoteRecord];
};
```

## Dataset Dihapus Handler

Ketika set data dihapus, Amazon Cognito client menggunakan `AWSCognitoDatasetDeletedHandler` untuk memastikan apakah salinan yang di-cache-kan secara lokal pada set data harus dihapus juga. Jika no `AWSCognitoDatasetDeletedHandler` tidak diimplementasikan, data lokal akan dibersihkan secara otomatis. Terapkan `AWSCognitoDatasetDeletedHandler` jika Anda ingin menyimpan salinan data lokal sebelum menghapus, atau untuk menyimpan data lokal.

Anda dapat mengatur handler dengan set data terhapus pada tingkat klien:

```
client.datasetDeletedHandler = ^BOOL (NSString *datasetName) {
    // make a backup of the data if you choose
    ...
    // delete the local data (default behavior)
    return YES;
};
```

Atau pada tingkat set data:

```
dataset.datasetDeletedHandler = ^BOOL (NSString *datasetName) {  
    // override default and keep the local data  
    return NO;  
};
```

## Penangan Gabungan Set Data

Ketika dua identitas yang sebelumnya tidak terhubung dihubungkan bersama-sama, semua set data mereka digabung. Aplikasi akan menerima notifikasi tentang penggabungan melalui `DatasetMergeHandler`. Handler akan menerima nama set data root serta array nama set data yang ditandai sebagai penggabungan dari set data root.

Jika no `DatasetMergeHandler` tidak diimplementasikan, set data ini akan diabaikan, tetapi akan terus menggunakan ruang dalam total set data maksimal 20 milik identitas.

Anda dapat mengatur handler dengan gabungan set data pada tingkat klien:

```
client.datasetMergedHandler = ^(NSString *datasetName, NSArray *datasets) {  
    // Blindly delete the datasets  
    for (NSString *name in datasets) {  
        AWSCognitoDataset *merged = [[AWSCognito defaultCognito]  
openOrCreateDataset:name];  
        [merged clear];  
        [merged synchronize];  
    }  
};
```

Atau pada tingkat set data:

```
dataset.datasetMergedHandler = ^(NSString *datasetName, NSArray *datasets) {  
    // Blindly delete the datasets  
    for (NSString *name in datasets) {  
        AWSCognitoDataset *merged = [[AWSCognito defaultCognito]  
openOrCreateDataset:name];  
        // do something with the data if it differs from existing dataset  
        ...  
        // now delete it  
        [merged clear];  
        [merged synchronize];  
    }  
};
```

## iOS - Swift

### Notifikasi Sinkronisasi

Amazon Cognito client akan memancarkan sejumlah peristiwa `NSNotification` selama panggilan sinkronisasi. Anda dapat mendaftar untuk memantau notifikasi ini melalui standar `NSNotificationCenter`:

```
NSNotificationCenter.defaultCenter().addObserver(observer: self,  
    selector: "myNotificationHandler",  
    name:NOTIFICATION_TYPE,  
    object:nil)
```

Amazon Cognito mendukung lima jenis notifikasi, sebagaimana yang tercantum di bawah ini.

#### `AWSCognitoDidStartSynchronizeNotification`

Dipanggil ketika operasi sinkronisasi dimulai. `userInfo` akan berisi set data kunci yang merupakan nama dari set data yang disinkronkan.

#### `AWSCognitoDidEndSynchronizeNotification`

Disebut ketika operasi sinkronisasi selesai (berhasil atau sebaliknya). `userInfo` akan berisi set data kunci yang merupakan nama dari set data yang disinkronkan.

#### `AWSCognitoDidFailToSynchronizeNotification`

Dipanggil ketika operasi sinkronisasi gagal. `userInfo` akan berisi set data kunci yang merupakan nama set data yang disinkronkan dan kesalahan kunci yang akan berisi kesalahan yang menyebabkan kegagalan.

#### `AWSCognitoDidChangeRemoteValueNotification`

Dipanggil ketika perubahan lokal berhasil didorong ke Amazon Cognito. `userInfo` akan berisi set data kunci yang merupakan nama set data yang disinkronkan dan kunci kunci yang akan berisi `NSArray` kunci record yang didorong.

#### `AWSCognitoDidChangeLocalValueFromRemoteNotification`

Dipanggil ketika nilai lokal berubah akibat operasi sinkronisasi. `userInfo` akan berisi set data kunci yang merupakan nama set data yang disinkronkan dan kunci kunci yang akan berisi `NSArray` kunci record yang telah berubah.

## Penangan Resolusi Konflik

Selama operasi sinkronisasi, konflik mungkin timbul jika kunci yang sama telah diubah di local store dan sync store. Jika Anda belum menetapkan penanganan resolusi konflik, Amazon Cognito menetapkan setelan default yaitu memilih pembaruan terbaru.

Dengan menerapkan dan menetapkan `AWSCognitoRecordConflictHandler` Anda dapat mengubah resolusi konflik default. Konflik parameter input `AWSCognitoConflict` berisi objek `AWSCognitoRecord` untuk data yang di-cache-kan secara lokal dan catatan yang bertentangan di sync store. Menggunakan `AWSCognitoConflict` Anda dapat menyelesaikan konflik dengan catatan lokal: [konflik `resolveWithLocal` Rekam], catatan jarak jauh: [konflik `resolveWithRemote` Rekam] atau nilai baru: [`resolveWithValueconflict:value`]. Menampilkan nil dari metode ini mencegah sinkronisasi berlanjut dan konflik akan ditampilkan lagi saat proses sinkronisasi dimulai kembali.

Anda dapat mengatur penanganan resolusi konflik di tingkat klien:

```
client.conflictHandler = {
    (datasetName: String?, conflict: AWSCognitoConflict?) ->
    AWSCognitoResolvedConflict? in
    return conflict.resolveWithLocalRecord()
}
```

Atau pada tingkat set data:

```
dataset.conflictHandler = {
    (datasetName: String?, conflict: AWSCognitoConflict?) ->
    AWSCognitoResolvedConflict? in
    return conflict.resolveWithLocalRecord()
}
```

## Dataset Dihapus Handler

Ketika set data dihapus, Amazon Cognito client menggunakan `AWSCognitoDatasetDeletedHandler` untuk memastikan apakah salinan yang di-cache-kan secara lokal pada set data harus dihapus juga. Jika no `AWSCognitoDatasetDeletedHandler` tidak diimplementasikan, data lokal akan dibersihkan secara otomatis. Terapkan `AWSCognitoDatasetDeletedHandler` jika Anda ingin menyimpan salinan data lokal sebelum menghapus, atau untuk menyimpan data lokal.

Anda dapat mengatur handler dengan set data terhapus pada tingkat klien:

```
client.datasetDeletedHandler = {
    (datasetName: String!) -> Bool in
    // make a backup of the data if you choose
    ...
    // delete the local data (default behaviour)
    return true
}
```

Atau pada tingkat set data:

```
dataset.datasetDeletedHandler = {
    (datasetName: String!) -> Bool in
    // make a backup of the data if you choose
    ...
    // delete the local data (default behaviour)
    return true
}
```

### Penangan gabungan set data

Ketika dua identitas yang sebelumnya tidak terhubung dihubungkan bersama-sama, semua set data mereka digabung. Aplikasi akan menerima notifikasi tentang penggabungan melalui `DatasetMergeHandler`. Handler akan menerima nama set data root serta array nama set data yang ditandai sebagai penggabungan dari set data root.

Jika no `DatasetMergeHandler` tidak diimplementasikan, set data ini akan diabaikan, tetapi akan terus menggunakan ruang dalam total set data maksimal 20 milik identitas.

Anda dapat mengatur handler dengan gabungan set data pada tingkat klien:

```
client.datasetMergedHandler = {
    (datasetName: String!, datasets: [AnyObject]!) -> Void in
    for nameObject in datasets {
        if let name = nameObject as? String {
            let merged = AWSCognito.defaultCognito().openOrCreateDataset(name)
            merged.clear()
            merged.synchronize()
        }
    }
}
```

Atau pada tingkat set data:

```
dataset.datasetMergedHandler = {
  (datasetName: String!, datasets: [AnyObject]!) -> Void in
  for nameObject in datasets {
    if let name = nameObject as? String {
      let merged = AWSCognito.defaultCognito().openOrCreateDataset(name)
      // do something with the data if it differs from existing dataset
      ...
      // now delete it
      merged.clear()
      merged.synchronize()
    }
  }
}
```

## JavaScript

### Callback sinkronisasi

Ketika melakukan `synchronize()` pada set data, Anda secara opsional dapat menentukan callback untuk menangani masing-masing status berikut:

```
dataset.synchronize({

  onSuccess: function(dataset, newRecords) {
    //...
  },

  onFailure: function(err) {
    //...
  },

  onConflict: function(dataset, conflicts, callback) {
    //...
  },

  onDatasetDeleted: function(dataset, datasetName, callback) {
    //...
  },

  onDatasetMerged: function(dataset, datasetNames, callback) {
    //...
  }
})
```



```
}  
  
});
```

### onSuccess ()

Callback `onSuccess()` dipicu saat set data berhasil diunggah dari sync store. Jika Anda tidak menentukan callback, sinkronisasi akan berhasil secara diam-diam.

```
onSuccess: function(dataset, newRecords) {  
    console.log('Successfully synchronized ' + newRecords.length + ' new records.');
```

### onFailure ()

`onFailure()` dipanggil jika pengecualian terjadi selama sinkronisasi. Jika Anda tidak menentukan callback, sinkronisasi akan gagal secara diam-diam.

```
onFailure: function(err) {  
    console.log('Synchronization failed.');
```

### padaKonflik ()

Konflik mungkin timbul jika kunci yang sama telah diubah di local store dan sync store. Metode `onConflict()` menangani resolusi konflik. Jika Anda tidak menerapkan metode ini, sinkronisasi akan dibatalkan jika terjadi konflik.

```
onConflict: function(dataset, conflicts, callback) {  
  
    var resolved = [];  
  
    for (var i=0; i<conflicts.length; i++) {  
  
        // Take remote version.  
        resolved.push(conflicts[i].resolveWithRemoteRecord());  
  
        // Or... take local version.  
        // resolved.push(conflicts[i].resolveWithLocalRecord());
```

```
// Or... use custom logic.
// var newValue = conflicts[i].getRemoteRecord().getValue() +
conflicts[i].getLocalRecord().getValue();
// resolved.push(conflicts[i].resovleWithValue(newValue);

}

dataset.resolve(resolved, function() {
  return callback(true);
});

// Or... callback false to stop the synchronization process.
// return callback(false);

}
```

### onDatasetDeleted()

Ketika set data dihapus, Amazon Cognito client menggunakan callback `onDatasetDeleted()` untuk memutuskan apakah salinan yang di-cache-kan secara lokal pada set data harus dihapus juga. Secara default, set data tidak akan dihapus.

```
onDatasetDeleted: function(dataset, datasetName, callback) {

  // Return true to delete the local copy of the dataset.
  // Return false to handle deleted datasets outside the synchronization callback.

  return callback(true);

}
```

### onDatasetMerged()

Ketika dua identitas yang sebelumnya tidak terhubung dihubungkan bersama-sama, semua set data mereka digabung. Aplikasi akan menerima notifikasi tentang penggabungan melalui callback `onDatasetsMerged()`.

```
onDatasetMerged: function(dataset, datasetNames, callback) {

  // Return true to continue the synchronization process.
  // Return false to handle dataset merges outside the synchronization callback.

}
```

```
return callback(false);  
  
}
```

## Unity

Setelah Anda membuka atau membuat set data, Anda dapat mengatur callback yang berbeda untuk itu yang akan dipicu ketika Anda menggunakan metode Sinkronisasi. Ini adalah cara untuk mendaftarkan callback Anda kepada mereka:

```
dataset.OnSyncSuccess += this.HandleSyncSuccess;  
dataset.OnSyncFailure += this.HandleSyncFailure;  
dataset.OnSyncConflict = this.HandleSyncConflict;  
dataset.OnDatasetMerged = this.HandleDatasetMerged;  
dataset.OnDatasetDeleted = this.HandleDatasetDeleted;
```

Perhatikan bahwa `SyncSuccess` dan `SyncFailure` menggunakan `+=` instead of `=` sehingga Anda dapat berlangganan lebih dari satu callback kepada mereka.

### OnSyncSuccess

Callback `OnSyncSuccess` dipicu saat set data berhasil diperbarui dari cloud. Jika Anda tidak menentukan callback, sinkronisasi akan berhasil secara diam-diam.

```
private void HandleSyncSuccess(object sender, SyncSuccessEvent e)  
{  
    // Continue with your game flow, display the loaded data, etc.  
}
```

### OnSyncFailure

`OnSyncFailure` dipanggil jika pengecualian terjadi selama sinkronisasi. Jika Anda tidak menentukan callback, sinkronisasi akan gagal secara diam-diam.

```
private void HandleSyncFailure(object sender, SyncFailureEvent e)  
{  
    Dataset dataset = sender as Dataset;  
    if (dataset.Metadata != null) {  
        Debug.Log("Sync failed for dataset : " + dataset.Metadata.DatasetName);  
    } else {
```

```
        Debug.Log("Sync failed");
    }
    // Handle the error
    Debug.LogException(e.Exception);
}
```

## OnSyncConflict

Konflik mungkin timbul jika kunci yang sama telah diubah di local store dan sync store. Callback `OnSyncConflict` menangani resolusi konflik. Jika Anda tidak menerapkan metode ini, sinkronisasi akan dibatalkan jika terjadi konflik.

```
private bool HandleSyncConflict(Dataset dataset, List < SyncConflict > conflicts)
{
    if (dataset.Metadata != null) {
        Debug.LogWarning("Sync conflict " + dataset.Metadata.DatasetName);
    } else {
        Debug.LogWarning("Sync conflict");
    }
    List < Amazon.CognitoSync.SyncManager.Record > resolvedRecords = new List <
Amazon.CognitoSync.SyncManager.Record > ();
    foreach(SyncConflict conflictRecord in conflicts) {
        // SyncManager provides the following default conflict resolution methods:
        //     ResolveWithRemoteRecord - overwrites the local with remote records
        //     ResolveWithLocalRecord - overwrites the remote with local records
        //     ResolveWithValue - to implement your own logic
        resolvedRecords.Add(conflictRecord.ResolveWithRemoteRecord());
    }
    // resolves the conflicts in local storage
    dataset.Resolve(resolvedRecords);
    // on return true the synchronize operation continues where it left,
    //     returning false cancels the synchronize operation
    return true;
}
```

## OnDatasetDeleted

Ketika set data dihapus, Amazon Cognito client menggunakan callback `OnDatasetDeleted` untuk memutuskan apakah salinan yang di-cache-kan secara lokal pada set data harus dihapus juga. Secara default, set data tidak akan dihapus.

```
private bool HandleDatasetDeleted(Dataset dataset)
```

```

{
    Debug.Log(dataset.Metadata.DatasetName + " Dataset has been deleted");
    // Do clean up if necessary
    // returning true informs the corresponding dataset can be purged in the local
    storage and return false retains the local dataset
    return true;
}

```

## OnDatasetMerged

Ketika dua identitas yang sebelumnya tidak terhubung dihubungkan bersama-sama, semua set data mereka digabung. Aplikasi akan menerima notifikasi tentang penggabungan melalui callback `OnDatasetsMerged`.

```

public bool HandleDatasetMerged(Dataset localDataset, List<string> mergedDatasetNames)
{
    foreach (string name in mergedDatasetNames)
    {
        Dataset mergedDataset = syncManager.OpenOrCreateDataset(name);
        //Lambda function to delete the dataset after fetching it
        EventHandler<SyncSuccessEvent> lambda;
        lambda = (object sender, SyncSuccessEvent e) => {
            ICollection<string> existingValues = localDataset.GetAll().Values;
            ICollection<string> newValues = mergedDataset.GetAll().Values;

            //Implement your merge logic here

            mergedDataset.Delete(); //Delete the dataset locally
            mergedDataset.OnSyncSuccess -= lambda; //We don't want this callback to be
            fired again
            mergedDataset.OnSyncSuccess += (object s2, SyncSuccessEvent e2) => {
                localDataset.Synchronize(); //Continue the sync operation that was
            interrupted by the merge
            };
            mergedDataset.Synchronize(); //Synchronize it as deleted, failing to do so
            will leave us in an inconsistent state
            };
            mergedDataset.OnSyncSuccess += lambda;
            mergedDataset.Synchronize(); //Asnchronously fetch the dataset
        }

        // returning true allows the Synchronize to continue and false stops it
        return false;
    }
}

```

```
}
```

## Xamarin

Setelah Anda membuka atau membuat set data, Anda dapat mengatur callback yang berbeda untuk itu yang akan dipicu ketika Anda menggunakan metode Sinkronisasi. Ini adalah cara untuk mendaftarkan callback Anda kepada mereka:

```
dataset.OnSyncSuccess += this.HandleSyncSuccess;  
dataset.OnSyncFailure += this.HandleSyncFailure;  
dataset.OnSyncConflict = this.HandleSyncConflict;  
dataset.OnDatasetMerged = this.HandleDatasetMerged;  
dataset.OnDatasetDeleted = this.HandleDatasetDeleted;
```

Perhatikan bahwa `SyncSuccess` dan `SyncFailure` menggunakan `+=` instead of `=` sehingga Anda dapat berlangganan lebih dari satu callback kepada mereka.

### OnSyncSuccess

Callback `OnSyncSuccess` dipicu saat set data berhasil diperbarui dari cloud. Jika Anda tidak menentukan callback, sinkronisasi akan berhasil secara diam-diam.

```
private void HandleSyncSuccess(object sender, SyncSuccessEventArgs e)  
{  
    // Continue with your game flow, display the loaded data, etc.  
}
```

### OnSyncFailure

`OnSyncFailure` dipanggil jika pengecualian terjadi selama sinkronisasi. Jika Anda tidak menentukan callback, sinkronisasi akan gagal secara diam-diam.

```
private void HandleSyncFailure(object sender, SyncFailureEventArgs e)  
{  
    Dataset dataset = sender as Dataset;  
    if (dataset.Metadata != null) {  
        Console.WriteLine("Sync failed for dataset : " + dataset.Metadata.DatasetName);  
    } else {  
        Console.WriteLine("Sync failed");  
    }  
}
```

## OnSyncConflict

Konflik mungkin timbul jika kunci yang sama telah diubah di local store dan sync store. Callback `OnSyncConflict` menangani resolusi konflik. Jika Anda tidak menerapkan metode ini, sinkronisasi akan dibatalkan jika terjadi konflik.

```
private bool HandleSyncConflict(Dataset dataset, List < SyncConflict > conflicts)
{
    if (dataset.Metadata != null) {
        Console.WriteLine("Sync conflict " + dataset.Metadata.DatasetName);
    } else {
        Console.WriteLine("Sync conflict");
    }
    List < Amazon.CognitoSync.SyncManager.Record > resolvedRecords = new List <
Amazon.CognitoSync.SyncManager.Record > ();
    foreach(SyncConflict conflictRecord in conflicts) {
        // SyncManager provides the following default conflict resolution methods:
        //     ResolveWithRemoteRecord - overwrites the local with remote records
        //     ResolveWithLocalRecord - overwrites the remote with local records
        //     ResolveWithValue - to implement your own logic
        resolvedRecords.Add(conflictRecord.ResolveWithRemoteRecord());
    }
    // resolves the conflicts in local storage
    dataset.Resolve(resolvedRecords);
    // on return true the synchronize operation continues where it left,
    //     returning false cancels the synchronize operation
    return true;
}
```

## OnDatasetDeleted

Ketika set data dihapus, Amazon Cognito client menggunakan callback `OnDatasetDeleted` untuk memutuskan apakah salinan yang di-cache-kan secara lokal pada set data harus dihapus juga. Secara default, set data tidak akan dihapus.

```
private bool HandleDatasetDeleted(Dataset dataset)
{
    Console.WriteLine(dataset.Metadata.DatasetName + " Dataset has been deleted");
    // Do clean up if necessary
    // returning true informs the corresponding dataset can be purged in the local
storage and return false retains the local dataset
    return true;
}
```

```
}
```

## OnDatasetMerged

Ketika dua identitas yang sebelumnya tidak terhubung dihubungkan bersama-sama, semua set data mereka digabung. Aplikasi akan menerima notifikasi tentang penggabungan melalui callback `OnDatasetsMerged`.


```
public bool HandleDatasetMerged(Dataset localDataset, List<string> mergedDatasetNames)
{
    foreach (string name in mergedDatasetNames)
    {
        Dataset mergedDataset = syncManager.OpenOrCreateDataset(name);

        //Implement your merge logic here

        mergedDataset.OnSyncSuccess += lambda;
        mergedDataset.SynchronizeAsync(); //Asnchronously fetch the dataset
    }

    // returning true allows the Synchronize to continue and false stops it
    return false;
}
```

## Sinkronisasi push

-  Jika Anda baru mengenal Amazon Cognito Sync, gunakan [AWS AppSync](#). Seperti Amazon Cognito Sync, AWS AppSync adalah layanan untuk menyinkronkan data aplikasi di seluruh perangkat. Hal ini memungkinkan data pengguna seperti preferensi aplikasi atau game state agar dapat disinkronkan. Hal ini juga memperluas kemampuan ini dengan memungkinkan beberapa pengguna untuk menyinkronkan dan berkolaborasi secara langsung pada data bersama.

Amazon Cognito secara otomatis melacak hubungan antara identitas dan perangkat. Dengan menggunakan fitur sinkronisasi push, atau push sync, Anda dapat memastikan bahwa setiap instans dari identitas yang diberikan akan diberitahukan ketika data identitas berubah. Sinkronisasi push memastikan bahwa, setiap kali sinkronisasi menyimpan perubahan data untuk identitas tertentu,



semua perangkat yang terkait dengan identitas tersebut menerima notifikasi push diam yang memberitahukan perubahan tersebut.

#### Note

Sinkronisasi push tidak didukung untuk JavaScript, Unity, atau Xamarin.

Sebelum Anda dapat menggunakan push sync, Anda harus terlebih dahulu menyiapkan akun Anda untuk push sync dan mengaktifkan push sync di konsol Amazon Cognito.

## Membuat aplikasi Amazon Simple Notification Service (Amazon SNS)

Buat dan konfigurasi aplikasi Amazon SNS untuk platform yang didukung, seperti yang dijelaskan dalam [Panduan Developer SNS](#).

## Aktifkan sinkronisasi push di konsol Amazon Cognito

Anda dapat mengaktifkan push sync di konsol Amazon Cognito. Dari [halaman beranda konsol](#):

1. Klik nama kolom identitas berisi push sync yang ingin Anda aktifkan. Halaman Dasbor untuk kolom identitas Anda akan muncul.
2. Di pojok kanan atas halaman Dasbor, klik Kelola Kolam Identitas. Halaman Gabungan identitas muncul.
3. Gulir ke bawah dan klik Sinkronisasi push untuk memperluasnya.
4. Di menu pilihan menurun Peran layanan, pilih IAM role yang memberikan izin pada Cognito untuk mengirim notifikasi SNS. Klik Buat peran untuk membuat atau mengubah peran yang terkait dengan kolom identitas Anda di [AWS Konsol IAM](#).
5. Pilih aplikasi platform, kemudian klik Simpan perubahan.
6. Berikan Akses SNS ke Aplikasi Anda

Di AWS Identity and Access Management konsol, konfigurasi peran IAM Anda untuk memiliki akses Amazon SNS penuh, atau buat peran baru yang memiliki akses Amazon SNS penuh. Contoh kebijakan kepercayaan peran berikut memberikan Amazon Cognito Sync kemampuan terbatas untuk mengambil peran IAM. Amazon Cognito Sync hanya dapat mengambil peran ketika melakukannya atas nama kumpulan identitas dalam `aws:SourceArn` kondisi dan akun dalam `aws:SourceAccount` kondisi tersebut.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "cognito-sync.amazonaws.com"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
          "AWS:SourceAccount": "123456789012"
        },
        "ArnLike": {
          "AWS:SourceArn": "arn:aws:cognito-identity:us-
east-1:123456789012:identitypool/us-east-1:177a950c-2c08-43f0-9983-28727EXAMPLE"
        }
      }
    }
  ]
}
```

Untuk belajar lebih lanjut tentang IAM role, lihat [Peran \(Delegasi dan Federasi\)](#).

## Menggunakan sinkronisasi push di aplikasi Anda: Android

Aplikasi Anda akan perlu mengimpor layanan Google Play. Anda dapat mengunduh versi terbaru Google Play SDK melalui [Pengelola SDK Android](#). Ikuti dokumentasi Android di [Implementasi Android](#) untuk mendaftarkan aplikasi Anda dan menerima ID pendaftaran dari GCM. Setelah Anda memiliki ID registrasi, Anda perlu mendaftarkan perangkat dengan Amazon Cognito, seperti yang ditunjukkan pada cuplikan di bawah ini:

```
String registrationId = "MY_GCM_REGISTRATION_ID";
try {
    client.registerDevice("GCM", registrationId);
} catch (RegistrationFailedException rfe) {
    Log.e(TAG, "Failed to register device for silent sync", rfe);
} catch (AmazonClientException ace) {
    Log.e(TAG, "An unknown error caused registration for silent sync to fail", ace);
}
```

Sekarang Anda dapat berlangganan perangkat untuk menerima pembaruan dari set data tertentu:

```
Dataset trackedDataset = client.openOrCreateDataset("myDataset");
if (client.isDeviceRegistered()) {
    try {
        trackedDataset.subscribe();
    } catch (SubscribeFailedException sfe) {
        Log.e(TAG, "Failed to subscribe to datasets", sfe);
    } catch (AmazonClientException ace) {
        Log.e(TAG, "An unknown error caused the subscription to fail", ace);
    }
}
```

Untuk berhenti menerima notifikasi push dari set data, cukup panggil metode berhenti berlangganan. Untuk berlangganan semua set data (atau subset tertentu) di objek `CognitoSyncManager`, gunakan `subscribeAll()`:

```
if (client.isDeviceRegistered()) {
    try {
        client.subscribeAll();
    } catch (SubscribeFailedException sfe) {
        Log.e(TAG, "Failed to subscribe to datasets", sfe);
    } catch (AmazonClientException ace) {
        Log.e(TAG, "An unknown error caused the subscription to fail", ace);
    }
}
```

Dalam implementasi `BroadcastReceiver` objek [Android](#), Anda dapat memeriksa versi terbaru dari kumpulan data yang dimodifikasi dan memutuskan apakah aplikasi Anda perlu melakukan sinkronisasi lagi:

```
@Override
public void onReceive(Context context, Intent intent) {

    PushSyncUpdate update = client.getPushSyncUpdate(intent);

    // The update has the source (cognito-sync here), identityId of the
    // user, identityPoolId in question, the non-local sync count of the
    // data set and the name of the dataset. All are accessible through
    // relevant getters.

    String source = update.getSource();
    String identityPoolId = update.getIdentityPoolId();
```

```
String identityId = update.getIdentityId();
String datasetName = update.getDatasetName();
long syncCount = update.getSyncCount();

Dataset dataset = client.openOrCreateDataset(datasetName);

// need to access last sync count. If sync count is less or equal to
// last sync count of the dataset, no sync is required.

long lastSyncCount = dataset.getLastSyncCount();
if (lastSyncCount < syncCount) {
    dataset.synchronize(new SyncCallback() {
        // ...
    });
}
}
```

Tombol berikut tersedia dalam muatan notifikasi push:

- **source: cognito-sync.** Ini dapat berfungsi sebagai faktor pembeda antara notifikasi.
- **identityPoolId:** ID kolam identitas. Ini dapat digunakan untuk validasi atau informasi tambahan, meskipun tidak terpisahkan dari sudut pandang penerima.
- **identityId:** ID identitas dalam kolam.
- **datasetName:** Nama set data yang diperbarui. Ini tersedia demi panggilan `openOrCreateDataset`.
- **syncCount:** Jumlah sinkronisasi untuk set data jarak jauh. Anda dapat menggunakan ini sebagai cara untuk memastikan bahwa set data lokal telah kedaluwarsa, dan sinkronisasi yang masuk adalah sinkronisasi baru.

## Menggunakan push sync di aplikasi Anda: iOS - Objective-C

Agar dapat mendapatkan token perangkat untuk aplikasi Anda, ikuti dokumentasi Apple pada Pendaftaran Notifikasi Jarak Jauh. Setelah Anda menerima token perangkat sebagai objek `NSData` dari APN, Anda harus mendaftarkan perangkat dengan Amazon Cognito dengan menggunakan metode klien sinkronisasi `registerDevice:`, seperti yang ditunjukkan di bagian bawah:

```
AWSCognito *syncClient = [AWSCognito defaultCognito];
[[syncClient registerDevice: devToken] continueWithBlock:^id(AWSTask *task) {
    if(task.error){
```

```

        NSLog(@"Unable to registerDevice: %@", task.error);
    } else {
        NSLog(@"Successfully registered device with id: %@", task.result);
    }
    return nil;
}
];

```

Dalam mode debug, perangkat Anda akan mendaftar dengan sandbox APN; dalam modus rilis, perangkat akan mendaftar dengan APN. Untuk menerima pembaruan dari set data tertentu, gunakan metode `subscribe`:

```

[[[syncClient openOrCreateDataset:@"MyDataset"] subscribe]
continueWithBlock:^id(AWSTask *task) {
    if(task.error){
        NSLog(@"Unable to subscribe to dataset: %@", task.error);
    } else {
        NSLog(@"Successfully subscribed to dataset: %@", task.result);
    }
    return nil;
}
];

```

Untuk berhenti menerima notifikasi push dari set data, cukup panggil metode `unsubscribe`:

```

[[[syncClient openOrCreateDataset:@"MyDataset"] unsubscribe]
continueWithBlock:^id(AWSTask *task) {
    if(task.error){
        NSLog(@"Unable to unsubscribe from dataset: %@", task.error);
    } else {
        NSLog(@"Successfully unsubscribed from dataset: %@", task.result);
    }
    return nil;
}
];

```

Untuk berlangganan semua set data di objek `AWSCognito`, panggil `subscribeAll`:

```

[[syncClient subscribeAll] continueWithBlock:^id(AWSTask *task) {
    if(task.error){
        NSLog(@"Unable to subscribe to all datasets: %@", task.error);
    } else {

```

```
        NSLog(@"Successfully subscribed to all datasets: %@", task.result);
    }
    return nil;
}
];
```

Sebelum memanggil `subscribeAll`, pastikan untuk menyinkronkan setidaknya sekali pada setiap set data, sehingga set data ada di server.

Untuk bereaksi terhadap notifikasi push, Anda perlu menerapkan metode `didReceiveRemoteNotification` di delegasi aplikasi Anda:

```
- (void)application:(UIApplication *)application didReceiveRemoteNotification:
(NSDictionary *)userInfo
{
    [[NSNotificationCenter defaultCenter]
    postNotificationName:@"CognitoPushNotification" object:userInfo];
}
```

Jika Anda mengirim notifikasi dengan menggunakan handler notifikasi, Anda kemudian dapat menanggapi notifikasi di tempat lain dalam aplikasi tempat Anda memiliki handler untuk set data Anda. Jika Anda berlangganan notifikasi seperti ini...

```
[[NSNotificationCenter defaultCenter] addObserver:self
selector:@selector(didReceivePushSync:)
name: :@"CognitoPushNotification" object:nil];
```

... Anda dapat bertindak atas notifikasi seperti ini:

```
- (void)didReceivePushSync:(NSNotification*)notification
{
    NSDictionary * data = [(NSDictionary *)[notification object]
    objectForKey:@"data"];
    NSString * identityId = [data objectForKey:@"identityId"];
    NSString * datasetName = [data objectForKey:@"datasetName"];
    if([self.dataset.name isEqualToString:datasetName] && [self.identityId
    isEqualToString:identityId]){
        [[self.dataset synchronize] continueWithBlock:^id(AWSTask *task) {
            if(!task.error){
                NSLog(@"Successfully synced dataset");
            }
        }
        return nil;
    }
```

```
        }];  
    }  
}
```

Tombol berikut tersedia dalam muatan notifikasi push:

- `source: cognito-sync`. Ini dapat berfungsi sebagai faktor pembeda antara notifikasi.
- `identityPoolId`: ID kolam identitas. Ini dapat digunakan untuk validasi atau informasi tambahan, meskipun tidak terpisahkan dari sudut pandang penerima.
- `identityId`: ID identitas dalam kolam.
- `datasetName`: Nama set data yang diperbarui. Ini tersedia untuk kepentingan panggilan `openOrCreateDataset`.
- `syncCount`: Jumlah sinkronisasi untuk set data jarak jauh. Anda dapat menggunakan ini sebagai cara untuk memastikan bahwa set data lokal telah kedaluwarsa, dan sinkronisasi yang masuk adalah sinkronisasi baru.

## Menggunakan push sync di aplikasi Anda: iOS - Swift

Agar dapat mendapatkan token perangkat untuk aplikasi Anda, ikuti dokumentasi Apple pada Pendaftaran Notifikasi Jarak Jauh. Setelah Anda menerima token perangkat sebagai objek `NSData` dari APN, Anda harus mendaftarkan perangkat dengan Amazon Cognito dengan menggunakan `RegisterDevice`: metode klien sinkronisasi, seperti yang ditunjukkan di bagian bawah:

```
let syncClient = AWSCognito.default()  
syncClient.registerDevice(devToken).continueWith(block: { (task: AWSTask!) ->  
    AnyObject! in  
    if (task.error != nil) {  
        print("Unable to register device: " + task.error.localizedDescription)  
    } else {  
        print("Successfully registered device with id: \(task.result)")  
    }  
    return task  
})
```

Dalam mode debug, perangkat Anda akan mendaftar dengan sandbox APN; dalam modus rilis, perangkat akan mendaftar dengan APN. Untuk menerima pembaruan dari set data tertentu, gunakan metode `subscribe`:

```
syncClient.openOrCreateDataset("MyDataset").subscribe().continueWith(block: { (task:
  AWSTask!) -> AnyObject! in
  if (task.error != nil) {
    print("Unable to subscribe to dataset: " + task.error.localizedDescription)

  } else {
    print("Successfully subscribed to dataset: \(task.result)")
  }
  return task
})
```

Untuk berhenti menerima notifikasi push dari set data, panggil metode unsubscribe:

```
syncClient.openOrCreateDataset("MyDataset").unsubscribe().continueWith(block: { (task:
  AWSTask!) -> AnyObject! in
  if (task.error != nil) {
    print("Unable to unsubscribe to dataset: " + task.error.localizedDescription)

  } else {
    print("Successfully unsubscribed to dataset: \(task.result)")
  }
  return task
})
```

Untuk berlangganan semua set data di objek AWSCognito, panggil subscribeAll:

```
syncClient.openOrCreateDataset("MyDataset").subscribeAll().continueWith(block: { (task:
  AWSTask!) -> AnyObject! in
  if (task.error != nil) {
    print("Unable to subscribe to all datasets: " + task.error.localizedDescription)

  } else {
    print("Successfully subscribed to all datasets: \(task.result)")
  }
  return task
})
```

Sebelum memanggil subscribeAll, pastikan untuk menyinkronkan setidaknya sekali pada setiap set data, sehingga set data ada di server.

Untuk bereaksi terhadap notifikasi push, Anda perlu menerapkan metode `didReceiveRemoteNotification` di delegasi aplikasi Anda:



```
func application(application: UIApplication, didReceiveRemoteNotification userInfo:
  [NSObject : AnyObject],
  fetchCompletionHandler completionHandler: (UIBackgroundFetchResult) -> Void) {

  NotificationCenter.defaultCenter().postNotificationName("CognitoPushNotification",
    object: userInfo)
}
```

Jika Anda mengirim notifikasi dengan menggunakan handler notifikasi, Anda kemudian dapat menanggapi notifikasi di tempat lain dalam aplikasi tempat Anda memiliki handler untuk set data Anda. Jika Anda berlangganan notifikasi seperti ini...

```
NotificationCenter.defaultCenter().addObserver(observer:self,
  selector:"didReceivePushSync:",
  name:"CognitoPushNotification",
  object:nil)
```

... Anda dapat bertindak atas notifikasi seperti ini:

```
func didReceivePushSync(notification: NSNotification) {
  if let data = (notification.object as! [String: AnyObject])["data"] as? [String:
  AnyObject] {
    let identityId = data["identityId"] as! String
    let datasetName = data["datasetName"] as! String


    if self.dataset.name == datasetName && self.identityId == identityId {
      dataset.synchronize().continueWithBlock {(task) -> AnyObject! in
        if task.error == nil {
          print("Successfully synced dataset")
        }
        return nil
      }
    }
  }
}
```

Tombol berikut tersedia dalam muatan notifikasi push:

- **source: cognito-sync.** Ini dapat berfungsi sebagai faktor pembeda antara notifikasi.
- **identityPoolId:** ID kolam identitas. Ini dapat digunakan untuk validasi atau informasi tambahan, meskipun tidak terpisahkan dari sudut pandang penerima.

- `identityId`: ID identitas dalam kolom.
- `datasetName`: Nama set data yang diperbarui. Ini tersedia untuk kepentingan panggilan `openOrCreateDataset`.
- `syncCount`: Jumlah sinkronisasi untuk set data jarak jauh. Anda dapat menggunakan ini sebagai cara untuk memastikan bahwa set data lokal telah kedaluwarsa, dan sinkronisasi yang masuk adalah sinkronisasi baru.

## Pengaliran Amazon Cognito

 Jika Anda baru mengenal Amazon Cognito Sync, gunakan [AWS AppSync](#). Seperti Amazon Cognito Sync, AWS AppSync adalah layanan untuk menyinkronkan data aplikasi di seluruh perangkat.

Hal ini memungkinkan data pengguna seperti preferensi aplikasi atau game state agar dapat disinkronkan. Hal ini juga memperluas kemampuan ini dengan memungkinkan beberapa pengguna untuk menyinkronkan dan berkolaborasi secara langsung pada data bersama.

Pengaliran Amazon Cognito memberikan developer kontrol dan wawasan terkait data mereka yang disimpan di Amazon Cognito. Developer sekarang dapat mengonfigurasi pengaliran Kinesis untuk menerima peristiwa saat data diperbarui dan disinkronkan. Amazon Cognito dapat mendorong setiap perubahan set data ke pengaliran Kinesis yang Anda miliki secara langsung.

Dengan menggunakan Amazon Cognito Streams, Anda dapat memindahkan semua data Sinkronisasi ke Kinesis, yang kemudian dapat dialirkan ke alat gudang data seperti Amazon Redshift untuk analisis lebih lanjut. Untuk belajar lebih lanjut tentang Kinesis, lihat [Memulai Dengan Menggunakan Amazon Kinesis](#).

### Mengkonfigurasi aliran

Anda dapat mengatur Amazon Cognito Streams di konsol Amazon Cognito. Untuk mengaktifkan Amazon Cognito Streams di konsol Amazon Cognito, Anda harus memilih pengaliran Kinesis yang akan dipublikasikan dan IAM role yang memberikan izin kepada Amazon Cognito untuk menempatkan peristiwa di pengaliran yang dipilih.

Dari [halaman beranda konsol](#):

1. Klik nama kolam identitas berisi Amazon Cognito Streams yang ingin Anda atur. Halaman Dasbor untuk kolam identitas Anda akan muncul.
2. Di pojok kanan atas halaman Dasbor, klik Kelola Kolam Identitas. Halaman Kelola Gabungan identitas muncul.
3. Gulir ke bawah dan klik Cognito Streams untuk memperluasnya.
4. Di menu pilihan menurun Nama pengaliran, pilih nama pengaliran Kinesis yang ada. Atau, klik Buat pengaliran untuk membuatnya, masukkan nama pengaliran dan jumlah serpihan. Untuk mempelajari serpihan dan untuk bantuan dalam memperkirakan jumlah serpihan yang diperlukan oleh pengaliran Anda, lihat [Panduan Developer Kinesis](#).
5. Di menu pilihan menurun Publikasikan peran, pilih IAM role yang memberikan izin kepada Amazon Cognito untuk memublikasikan pengaliran Anda. Klik Buat peran untuk membuat atau mengubah peran yang terkait dengan kolam identitas Anda di [AWS Konsol IAM](#).
6. Di menu pilihan menurun Status pengaliran, pilih Diaktifkan untuk mengaktifkan pembaruan pengaliran. Klik Simpan perubahan.

Setelah Anda berhasil mengkonfigurasi pengaliran Amazon Cognito, semua pembaruan berikutnya untuk set data di kolam identitas ini akan dikirim ke pengaliran.

### Streaming konten

Setiap catatan yang dikirim ke pengaliran merupakan sinkronisasi tunggal. Berikut adalah contoh dari catatan yang dikirim ke pengaliran:

```
{
  "identityPoolId": "Pool Id",
  "identityId": "Identity Id",
  "dataSetName": "Dataset Name",
  "operation": "(replace|remove)",
  "kinesisSyncRecords": [
    {
      "key": "Key",
      "value": "Value",
      "syncCount": 1,
      "lastModifiedDate": 1424801824343,
      "deviceLastModifiedDate": 1424801824343,
      "op": "(replace|remove)"
    },
    ...
  ]
}
```

```
],  
  "lastModifiedDate": 1424801824343,  
  "kinesisSyncRecordsURL": "S3Url",  
  "payloadType": "(S3Url|Inline)",  
  "syncCount": 1  
}
```

Untuk pembaruan yang lebih besar dari ukuran payload maksimum Kinesis 1 MB, Amazon Cognito memasukkan URL Amazon S3 yang telah ditandatangani sebelumnya yang berisi konten lengkap pembaruan.

Setelah Anda mengonfigurasi aliran Amazon Cognito, jika Anda menghapus aliran Kinesis atau mengubah izin kepercayaan peran sehingga Amazon Cognito Sync tidak dapat lagi mengambil peran, Anda mematikan aliran Amazon Cognito. Anda harus membuat ulang aliran Kinesis atau memperbaiki peran, dan kemudian Anda harus mengaktifkan aliran lagi.


### Penerbitan massal

Setelah Anda mengkonfigurasi pengaliran Amazon Cognito, Anda akan dapat menjalankan operasi publikasi massal untuk data yang ada di kolam identitas Anda. Setelah Anda memulai operasi publikasi massal, baik melalui konsol atau langsung melalui API, Amazon Cognito akan mulai memublikasikan data ini ke pengaliran yang sama yang menerima pembaruan Anda.

Amazon Cognito tidak menjamin keunikan data yang dikirim ke pengaliran saat menggunakan operasi publikasi massal. Anda mungkin menerima pembaruan yang sama baik sebagai pembaruan maupun sebagai bagian dari publikasi massal. Ingatlah hal ini saat memproses catatan dari stream Anda.

Untuk memublikasikan semua pengaliran secara massal, ikuti langkah 1-6 di bawah Mengkonfigurasi Pengaliran, lalu klik Mulai publikasi massal. Anda hanya boleh melakukan satu operasi publikasi massal berjalan pada waktu tertentu dan mendapatkan satu permintaan publikasi massal yang sukses setiap 24 jam.

## Peristiwa Amazon Cognito

 Jika Anda baru mengenal Amazon Cognito Sync, gunakan [AWS AppSync](#). Seperti Amazon Cognito Sync, AWS AppSync adalah layanan untuk menyinkronkan data aplikasi di seluruh perangkat.

Hal ini memungkinkan data pengguna seperti preferensi aplikasi atau game state agar dapat disinkronkan. Hal ini juga memperluas kemampuan ini dengan memungkinkan beberapa pengguna untuk menyinkronkan dan berkolaborasi secara langsung pada data bersama.

Peristiwa Amazon Cognito memungkinkan Anda untuk mengeksekusi fungsi AWS Lambda dalam menanggapi peristiwa penting di Amazon Cognito. Amazon Cognito memunculkan peristiwa Sync Trigger ketika set data disinkronkan. Anda dapat menggunakan peristiwa Sync Trigger untuk mengambil tindakan ketika pengguna memperbarui data. Fungsi ini dapat mengevaluasi dan secara opsional memanipulasi data sebelum disimpan di cloud dan disinkronkan ke perangkat lain pengguna. Hal ini berguna untuk memvalidasi data yang berasal dari perangkat sebelum disinkronkan ke perangkat pengguna lain, atau untuk memperbarui nilai lain dalam set data berdasarkan data yang masuk seperti mengeluarkan penghargaan saat pemain mencapai tingkat yang baru.

Langkah-langkah di bawah ini akan memandu Anda melalui penyiapan fungsi Lambda yang mengeksekusi setiap kali Amazon Cognito Dataset disinkronkan.

#### Note

Bila menggunakan peristiwa Amazon Cognito, Anda hanya dapat menggunakan kredensial yang diperoleh dari Amazon Cognito Identity. Jika Anda memiliki fungsi Lambda terkait, tetapi Anda memanggil UpdateRecords dengan kredensial akun AWS (kredensial developer), fungsi Lambda Anda tidak akan dipanggil.

## Membuat fungsi di AWS Lambda

Untuk mengintegrasikan Lambda dengan Amazon Cognito, Anda harus terlebih dahulu membuat fungsi di Lambda. Untuk melakukannya:

### Memilih Fungsi Lambda di Amazon Cognito

1. Buka konsol Lambda.
2. Klik Buat Fungsi Lambda.
3. Di layar Pilih cetak biru, cari dan pilih "cognito-sync-trigger."
4. Pada layar Konfigurasi sumber acara, biarkan Jenis sumber peristiwa disetel ke "Cognito Sync Trigger" dan pilih kolam identitas Anda. Klik Berikutnya.

**Note**

Saat mengonfigurasi pemacu Amazon Cognito Sync di luar konsol, Anda harus menambahkan izin berbasis sumber daya Lambda untuk memungkinkan Amazon Cognito menjalankan fungsi tersebut. Anda dapat menambahkan izin ini dari konsol Lambda (lihat [Menggunakan kebijakan berbasis sumber daya untuk AWS Lambda](#)) atau dengan menggunakan operasi Lambda. [AddPermission](#)

Contoh Kebijakan Berbasis Sumber Daya Lambda

Kebijakan AWS Lambda berbasis sumber daya berikut memberi Amazon Cognito kemampuan terbatas untuk menjalankan fungsi Lambda. Amazon Cognito hanya dapat memanggil fungsi atas nama kumpulan identitas dalam `aws:SourceArn` kondisi dan akun dalam kondisi tersebut `aws:SourceAccount`.

```
{
  "Version": "2012-10-17",
  "Id": "default",
  "Statement": [
    {
      "Sid": "lambda-allow-cognito-my-function",
      "Effect": "Allow",
      "Principal": {
        "Service": "cognito-sync.amazonaws.com"
      },
      "Action": "lambda:InvokeFunction",
      "Resource": "<your Lambda function ARN>",
      "Condition": {
        "StringEquals": {
          "AWS:SourceAccount": "<your account number>"
        },
        "ArnLike": {
          "AWS:SourceArn": "<your identity pool ARN>"
        }
      }
    }
  ]
}
```

5. Di layar fungsi Konfigurasi, masukkan nama dan deskripsi untuk fungsi Anda. Biarkan Waktu Aktif disetel ke "Node.js." Biarkan kode tidak berubah untuk contoh kita. Contoh default tidak membuat

perubahan pada data yang disinkronkan. Ini hanya mencatat fakta bahwa peristiwa Amazon Cognito Sync Trigger terjadi. Biarkan nama Handler disetel ke "index.handler." Untuk Peran, pilih IAM role yang memberikan izin kode kepada Anda untuk mengakses AWS Lambda. Untuk mengubah peran, lihat konsol IAM. Biarkan pengaturan Lanjutan tidak berubah. Klik Berikutnya.

6. Pada layar Ulasan, tinjau perincian dan klik Buat fungsi. Halaman berikutnya menampilkan fungsi Lambda baru Anda.

Setelah Anda memiliki fungsi yang sesuai yang ditulis dalam Lambda, Anda harus memilih fungsi itu sebagai handler untuk acara Amazon Cognito Sync Trigger. Langkah-langkah di bawah ini memandu Anda melalui proses ini.

Dari halaman beranda konsol:

1. Klik nama kolam identitas berisi Amazon Cognito Events yang ingin Anda atur. Halaman Dasbor untuk kolam identitas Anda akan muncul.
2. Di pojok kanan atas halaman Dasbor, klik Kelola Identitas Gabungan. Halaman Kelola Gabungan identitas muncul.
3. Gulir ke bawah dan klik Cognito Events untuk memperluasnya.
4. Di menu pilihan menurun Sync Trigger, pilih fungsi Lambda yang ingin Anda picu saat peristiwa Sinkronisasi terjadi.
5. Klik Simpan perubahan.

Sekarang, fungsi Lambda Anda akan dieksekusi setiap kali set data disinkronkan. Bagian selanjutnya menjelaskan bagaimana Anda dapat membaca dan memodifikasi data dalam fungsi Anda karena sedang disinkronkan.

### Menulis fungsi Lambda untuk pemicu sinkronisasi

Pemicu sinkronisasi mengikuti pola pemrograman yang digunakan antarmuka penyedia layanan. Amazon Cognito memberikan masukan ke fungsi Lambda Anda dalam format JSON berikut.

```
{
  "version": 2,
  "eventType": "SyncTrigger",
  "region": "us-east-1",
  "identityPoolId": "identityPoolId",
  "identityId": "identityId",
```

```
"datasetName": "datasetName",
"datasetRecords": {
  "SampleKey1": {
    "oldValue": "oldValue1",
    "newValue": "newValue1",
    "op": "replace"
  },
  "SampleKey2": {
    "oldValue": "oldValue2",
    "newValue": "newValue2",
    "op": "replace"
  },...
}
}
```

Amazon Cognito mengharapkan nilai pengembalian fungsi memiliki format yang sama dengan input.

Saat Anda menulis fungsi untuk peristiwa Pemicu Sinkronisasi, amati hal berikut:

- Saat Amazon Cognito memanggil fungsi Lambda Anda selama waktuUpdateRecords, fungsi harus merespons dalam waktu 5 detik. Jika tidak, layanan Amazon Cognito Sync menghasilkan pengecualian. LambdaSocketTimeoutException Anda tidak dapat meningkatkan nilai batas waktu ini.
- Jika Anda mendapatkan LambdaThrottledException pengecualian, coba operasi sinkronisasi lagi untuk memperbarui catatan.
- Amazon Cognito menyediakan semua catatan yang ada dalam kumpulan data sebagai input ke fungsi.
- Rekaman bahwa pengguna aplikasi diperbarui memiliki op bidang yang ditetapkan sebagaireplace. Catatan dihapus memiliki op bidang ditetapkan sebagairemove.
- Anda dapat memodifikasi rekaman apa pun, meskipun pengguna aplikasi tidak memperbarui rekaman.
- Semua bidang kecuali DatasetRecords adalah read-only. Jangan mengubahnya. Jika Anda mengubah bidang ini, Anda tidak dapat memperbarui catatan.
- Untuk memodifikasi nilai catatan, memperbarui nilai dan mengatur op ke replace.
- Untuk menghapus catatan, baik mengatur op ke remove, atau menetapkan nilai ke null.
- Untuk menambahkan catatan, tambahkan catatan baru ke array DatasetRecords.
- Amazon Cognito mengabaikan catatan yang dihilangkan dalam respons saat Amazon Cognito updates catatan.



## Contoh fungsi Lambda

Contoh fungsi Lambda berikut menunjukkan cara mengakses, memodifikasi, dan menghapus data.

```
console.log('Loading function');

exports.handler = function(event, context) {
    console.log(JSON.stringify(event, null, 2));

    //Check for the event type
    if (event.eventType === 'SyncTrigger') {

        //Modify value for a key
        if('SampleKey1' in event.datasetRecords){
            event.datasetRecords.SampleKey1.newValue = 'ModifyValue1';
            event.datasetRecords.SampleKey1.op = 'replace';
        }

        //Remove a key
        if('SampleKey2' in event.datasetRecords){
            event.datasetRecords.SampleKey2.op = 'remove';
        }

        //Add a key
        if(!('SampleKey3' in event.datasetRecords)){
            event.datasetRecords.SampleKey3={'newValue':'ModifyValue3', 'op' :
'replace'};
        }
    }
    context.done(null, event);
};
```

# Menggunakan konsol Amazon Cognito

Anda dapat menggunakan [Konsol Amazon Cognito](#) untuk membuat dan mengelola kolam pengguna dan kolam identitas.

Panduan ini menyediakan step-by-step penelusuran untuk tugas kumpulan pengguna Amazon Cognito umum di konsol Amazon Cognito.

Untuk menggunakan konsol Amazon Cognito

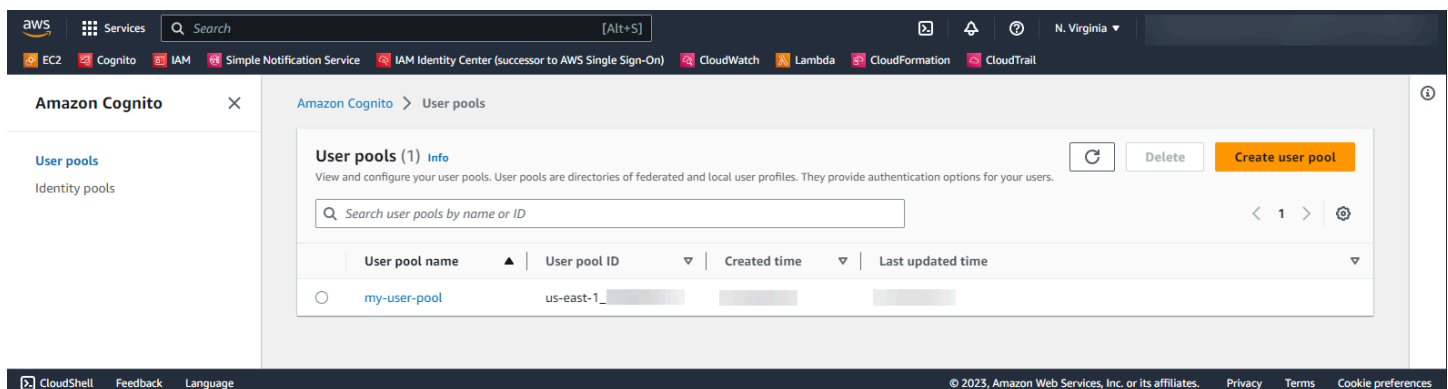
1. Untuk menggunakan Amazon Cognito, Anda harus [mendaftar akun AWS](#).
2. Masuk ke [Konsol Amazon Cognito](#). Anda mungkin diminta untuk AWS kredensial Anda.
3. Untuk membuat atau mengedit kumpulan pengguna, pilih Kumpulan Pengguna dari panel navigasi kiri.

Untuk informasi selengkapnya, lihat [Memulai dengan kumpulan pengguna](#).

4. Untuk membuat atau mengedit kumpulan identitas, pilih Identity pool. Anda akan diarahkan ke konsol asli untuk kumpulan identitas Amazon Cognito.

Untuk informasi selengkapnya, lihat [Memulai dengan kumpulan identitas Amazon Cognito](#).

Konsol Amazon Cognito adalah bagian dari AWS Management Console, yang menyediakan informasi tentang akun dan penagihan Anda. Untuk informasi lebih lanjut, lihat [Bekerja dengan AWS Management Console](#).



## Topik

- [Konsol kolam pengguna](#)
- [Konsol kumpulan identitas](#)

# Konsol kolam pengguna

Dari tampilan kumpulan Pengguna di konsol Amazon Cognito, pilih kumpulan pengguna dari daftar untuk melihat detail. Dalam tampilan mendetail, ikhtisar kumpulan pengguna di bagian atas konsol berisi informasi dasar tentang kumpulan pengguna Anda. Tab berikut mengatur konfigurasi kumpulan pengguna Anda ke dalam fungsi terkait.

## Pengguna

Tab Pengguna berisi informasi tentang pengguna dan impor pengguna dari file CSV. Anda dapat menambahkan, menghapus, dan mengedit pengguna di tab ini.

### Referensi

- [Mengelola pengguna di kumpulan pengguna Anda](#)
- [Mengimpor pengguna ke kumpulan pengguna dari file CSV](#)

## Grup

Tab Grup berisi informasi tentang grup pengguna. Anda dapat menambahkan, memodifikasi, dan mengubah keanggotaan dalam grup, dan mengubah peran IAM yang terkait dengan grup untuk integrasi kumpulan identitas.

### Referensi

- [Menambahkan grup ke kumpulan pengguna](#)

## Pengalaman masuk

Tab Pengalaman masuk berisi informasi tentang cara pengguna masuk ke kumpulan pengguna Anda. Di tab ini terdapat penyedia identitas pihak ketiga, opsi nama pengguna, kebijakan kata sandi, konfigurasi otentikasi multi-faktor (MFA), perilaku lupa kata sandi, dan mengingat perangkat. Anda dapat menambahkan dan memodifikasi penyedia identitas, dan mengubah keseluruhan perilaku masuk kumpulan pengguna Anda.

### Referensi

- [Menambahkan login kumpulan pengguna melalui pihak ketiga](#)
- [Menyesuaikan atribut masuk](#)
- [Menambahkan persyaratan kata sandi kumpulan pengguna](#)
- [Menambahkan MFA ke kumpulan pengguna](#)

- [Memulihkan akun pengguna](#)
- [Bekerja dengan perangkat pengguna di kumpulan pengguna Anda](#)

## Pengalaman pendaftaran

Tab Pengalaman pendaftaran berisi informasi tentang pendaftaran layanan mandiri, atribut yang diperlukan, memverifikasi nomor telepon dan alamat email, dan atribut khusus.

### Referensi

- [Mendaftar dan mengonfirmasi akun pengguna](#)
- [Atribut kumpulan pengguna](#)
- [Memverifikasi informasi kontak saat mendaftar](#)

## Perpesanan

Tab Pesan berisi informasi tentang Layanan AWS yang ingin Anda gunakan untuk mengirim email dan pesan SMS ke pengguna Anda, dan format pesan yang ingin Anda kirimkan kepada mereka.

### Referensi

- [Pengaturan email untuk kumpulan pengguna Amazon Cognito](#)
- [Pengaturan pesan SMS untuk kolam pengguna Amazon Cognito](#)
- [Mengkonfigurasi pesan verifikasi SMS dan email dan pesan undangan pengguna](#)

## Integrasi aplikasi

Tab Integrasi aplikasi berisi informasi tentang klien aplikasi kumpulan pengguna, domain yang Anda tetapkan ke titik akhir layanan kumpulan pengguna, server sumber daya API, UI yang dihosting, dan keamanan lanjutan. Anda dapat menelusuri setiap klien aplikasi untuk mengonfigurasi hal berikut.

1. Pengaturan token
2. URL panggilan balik
3. Alur otentikasi
4. Izin atribut
5. Keamanan lanjutan khusus aplikasi dan pengaturan UI yang dihosting
6. Amazon Pinpoint analitik

### Referensi

- [Klien aplikasi kumpulan pengguna](#)

- [Menyiapkan dan menggunakan UI yang dihosting Amazon Cognito dan titik akhir federasi](#)
- [Mengkonfigurasi domain kolam pengguna](#)
- [Cakupan, M2M, dan otorisasi API dengan server sumber daya](#)
- [Menambahkan keamanan tingkat lanjut ke kumpulan pengguna](#)
- [Menggunakan analitik Amazon Pinpoint dengan kumpulan pengguna Amazon Cognito](#)

## Properti kolam pengguna

Tab Properti kumpulan pengguna berisi informasi tentang konfigurasi kumpulan pengguna yang tidak terkait langsung dengan pengguna: Pemicu Lambda, perlindungan ACL AWS WAF web, perlindungan penghapusan, dan tag sumber daya.

### Referensi

- [Menyesuaikan alur kerja kumpulan pengguna dengan pemicu Lambda](#)
- [Menggaitkan ACL AWS WAF web dengan kumpulan pengguna](#)
- [perlindungan penghapusan kolam pengguna](#)
- [Menandai sumber daya Anda AWS](#)

## Konsol kumpulan identitas

Dari tampilan kumpulan Identitas di konsol Amazon Cognito, pilih kumpulan identitas dari daftar untuk melihat detail. Dalam tampilan mendetail, ikhtisar kumpulan Identitas di bagian atas konsol berisi informasi dasar tentang kumpulan pengguna Anda. Tab berikut mengatur konfigurasi kumpulan pengguna Anda ke dalam fungsi terkait.

### Statistik pengguna

Tab Statistik pengguna menampilkan informasi statistik tentang pengguna yang telah menghasilkan identitas di kumpulan identitas Anda. Anda tidak dapat mengonfigurasi pengaturan kumpulan identitas apa pun di tab ini.

### Browser identitas

Tab Identity Browser berisi informasi tentang identitas individual yang telah dihasilkan pengguna di kumpulan identitas Anda. Anda dapat melihat dan menghapus identitas.

### Referensi

- [Memulai dengan kumpulan identitas Amazon Cognito](#)

## Akses pengguna

Tab Akses pengguna berisi informasi tentang penyedia identitas yang telah Anda tautkan ke kumpulan identitas, penyedia pengembang, peran IAM default yang ditetapkan ke identitas, dan konfigurasi akses tamu yang tidak diautentikasi. Anda dapat menelusuri setiap penyedia identitas untuk mengonfigurasi hal berikut.

1. Kontrol akses berbasis peran dengan pemilihan Peran IAM
2. Kontrol akses berbasis atribut dengan Atribut untuk kontrol akses

### Referensi

- [Identitas mengumpulkan penyedia identitas eksternal](#)
- [Peran IAM](#)
- [Identitas yang diautentikasi dan tidak diautentikasi](#)
- [Identitas yang diautentikasi pengembang \(kumpulan identitas\)](#)
- [Menggunakan kontrol akses berbasis peran](#)
- [Menggunakan atribut untuk kontrol akses](#)

### Properti kolam identitas

Tab properti kumpulan identitas berisi informasi tentang konfigurasi kumpulan identitas lain-lain: otentikasi dasar (klasik) dan tag sumber daya.

- [Aliran otentikasi kumpulan identitas \(identitas federasi\)](#)
- [Menandai sumber daya Anda AWS](#)

# Keamanan di Amazon Cognito

Keamanan cloud di AWS adalah prioritas tertinggi. Sebagai AWS pelanggan, Anda mendapat manfaat dari pusat data dan arsitektur jaringan yang dibangun untuk memenuhi persyaratan organisasi yang paling sensitif terhadap keamanan.

Keamanan adalah tanggung jawab bersama antara Anda AWS dan Anda. [Model tanggung jawab bersama](#) menjelaskan hal ini sebagai keamanan cloud dan keamanan dalam cloud:

- Keamanan cloud — AWS bertanggung jawab untuk melindungi infrastruktur yang menjalankan AWS layanan di AWS Cloud. AWS juga memberi Anda layanan yang dapat Anda gunakan dengan aman. Auditor pihak ketiga secara teratur menguji dan memverifikasi efektivitas keamanan kami sebagai bagian dari [Program AWS Kepatuhan Program AWS Kepatuhan](#) . Untuk mempelajari tentang program kepatuhan yang berlaku untuk Amazon Cognito, lihat [AWS Layanan dalam Lingkup berdasarkan AWS Layanan Program Kepatuhan dalam Lingkup oleh Program](#) .
- Keamanan di cloud — Tanggung jawab Anda ditentukan oleh AWS layanan yang Anda gunakan. Anda juga bertanggung jawab atas faktor lain, yang mencakup kepekaan data Anda, persyaratan perusahaan, serta peraturan perundangan yang berlaku

Dokumentasi ini membantu Anda memahami cara menerapkan model tanggung jawab bersama saat menggunakan Amazon Cognito. Topik berikut menunjukkan kepada Anda cara mengonfigurasi Amazon Cognito untuk memenuhi tujuan keamanan dan kepatuhan Anda. Anda juga mempelajari cara menggunakan AWS layanan lain yang membantu Anda memantau dan mengamankan sumber daya Amazon Cognito Anda.

## Daftar Isi

- [Perlindungan data di Amazon Cognito](#)
- [Manajemen identitas dan akses untuk Amazon Cognito](#)
- [Pencatatan dan pemantauan di Amazon Cognito](#)
- [Validasi kepatuhan untuk Amazon Cognito](#)
- [Ketahanan di Amazon Cognito](#)
- [Keamanan infrastruktur di Amazon Cognito](#)
- [Analisis konfigurasi dan kerentanan di kumpulan pengguna Amazon Cognito](#)
- [AWS kebijakan terkelola untuk Amazon Cognito](#)

# Perlindungan data di Amazon Cognito

[Model tanggung jawab AWS bersama model](#) berlaku untuk perlindungan data di Amazon Cognito (Amazon Cognito). Seperti yang dijelaskan dalam model AWS ini, bertanggung jawab untuk melindungi infrastruktur global yang menjalankan semua AWS Cloud. Anda bertanggung jawab untuk mempertahankan kendali atas konten yang di-host pada infrastruktur ini. Konten ini mencakup konfigurasi keamanan dan tugas manajemen untuk AWS layanan yang Anda gunakan. Untuk informasi selengkapnya tentang privasi data, silakan lihat [Pertanyaan Umum Privasi Data](#).

Untuk tujuan perlindungan data, kami menyarankan Anda melindungi kredensi AWS akun dan menyiapkan akun pengguna individu dengan AWS Identity and Access Management (IAM). Dengan cara ini, setiap pengguna hanya diberi izin yang diperlukan untuk memenuhi tanggung jawab tugas mereka. Kami juga merekomendasikan agar Anda mengamankan data Anda dengan cara-cara berikut ini:

- Gunakan autentikasi multi-faktor (MFA) pada setiap akun.
- Gunakan SSL/TLS untuk berkomunikasi dengan sumber daya. AWS
- Siapkan API dan pencatatan aktivitas pengguna dengan AWS CloudTrail.
- Gunakan solusi AWS enkripsi, bersama dengan semua kontrol keamanan default dalam AWS layanan.
- Gunakan layanan keamanan terkelola lanjutan seperti Amazon Macie, yang membantu menemukan dan mengamankan data pribadi yang disimpan di Amazon S3.

Sebaiknya jangan pernah memasukkan informasi identitas yang sensitif, seperti nomor rekening pelanggan Anda, ke dalam bidang isian bebas seperti bidang Nama. Ini termasuk saat Anda bekerja dengan Amazon Cognito atau AWS layanan lain menggunakan konsol, API AWS CLI, atau AWS SDK. Setiap data yang Anda masukkan ke dalam Amazon Cognito atau layanan lain mungkin akan diambil untuk disertakan dalam log diagnostik. Saat Anda memberikan URL ke server eksternal, jangan menyertakan informasi kredensial dalam URL untuk memvalidasi permintaan Anda ke server tersebut.

## Enkripsi data

Enkripsi data biasanya terbagi dalam dua kategori: enkripsi saat diam dan enkripsi dalam transit.

### Enkripsi saat istirahat



Data dalam Amazon Cognito dienkripsi saat istirahat sesuai dengan standar industri.

## Enkripsi dalam perjalanan

Sebagai layanan terkelola, Amazon Cognito dilindungi oleh keamanan jaringan AWS global. Untuk informasi tentang layanan AWS keamanan dan cara AWS melindungi infrastruktur, lihat [Keamanan AWS Cloud](#). Untuk mendesain AWS lingkungan Anda menggunakan praktik terbaik untuk keamanan infrastruktur, lihat [Perlindungan Infrastruktur dalam Kerangka Kerja](#) yang AWS Diarsiteksikan dengan Baik Pilar Keamanan.

Anda menggunakan panggilan API yang AWS dipublikasikan untuk mengakses Amazon Cognito melalui jaringan. Klien harus mendukung hal-hal berikut:

- Keamanan Lapisan Pengangkutan (TLS). Kami mensyaratkan TLS 1.2 dan menganjurkan TLS 1.3.
- Sandi cocok dengan sistem kerahasiaan maju sempurna (perfect forward secrecy, PFS) seperti DHE (Ephemeral Diffie-Hellman) atau ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Sebagian besar sistem modern seperti Java 7 dan versi lebih baru mendukung mode-mode ini.

Selain itu, permintaan harus ditandatangani menggunakan ID kunci akses dan kunci akses rahasia yang terkait dengan prinsipal IAM. Atau Anda dapat menggunakan [AWS Security Token Service](#) (AWS STS) untuk menghasilkan kredensial keamanan sementara untuk menandatangani permintaan.

Kumpulan pengguna dan kumpulan identitas Amazon Cognito memiliki operasi API yang diautentikasi, tidak diautentikasi, dan diotorisasi token oleh IAM. Operasi API yang tidak diautentikasi dan diberi otorisasi token dimaksudkan untuk digunakan oleh pelanggan Anda, pengguna akhir aplikasi Anda. Operasi API yang tidak diautentikasi dan diotorisasi token dienkripsi saat istirahat dan dalam perjalanan. Untuk informasi selengkapnya, lihat [Kumpulan pengguna Amazon Cognito operasi API yang diautentikasi dan tidak diautentikasi](#).

### Note

Amazon Cognito mengenkripsi konten Anda secara internal dan tidak mendukung kunci yang disediakan pelanggan.

## Manajemen identitas dan akses untuk Amazon Cognito

AWS Identity and Access Management (IAM) adalah Layanan AWS yang membantu administrator mengontrol akses ke AWS sumber daya dengan aman. Administrator IAM mengontrol siapa yang dapat diautentikasi (masuk) dan diotorisasi (memiliki izin) untuk menggunakan sumber daya Amazon Cognito. IAM adalah Layanan AWS yang dapat Anda gunakan tanpa biaya tambahan.

## Topik

- [Audiens](#)
- [Mengautentikasi dengan identitas](#)
- [Mengelola akses menggunakan kebijakan](#)
- [Cara kerja Amazon Cognito dengan IAM](#)
- [Contoh kebijakan berbasis identitas untuk Amazon Cognito](#)
- [Pemecahan masalah identitas dan akses Amazon Cognito](#)
- [Menggunakan peran terkait layanan untuk Amazon Cognito](#)

## Audiens

Cara Anda menggunakan AWS Identity and Access Management (IAM) berbeda, tergantung pada pekerjaan yang Anda lakukan di Amazon Cognito.

Pengguna layanan — Jika Anda menggunakan layanan Amazon Cognito untuk melakukan pekerjaan Anda, administrator Anda akan memberi Anda kredensi dan izin yang Anda butuhkan. Saat Anda menggunakan lebih banyak fitur Amazon Cognito untuk melakukan pekerjaan Anda, Anda mungkin memerlukan izin tambahan. Memahami cara akses dikelola dapat membantu Anda meminta izin yang tepat dari administrator Anda. Jika Anda tidak dapat mengakses fitur di Amazon Cognito, lihat. [Pemecahan masalah identitas dan akses Amazon Cognito](#)

Administrator layanan - Jika Anda bertanggung jawab atas sumber daya Amazon Cognito di perusahaan Anda, Anda mungkin memiliki akses penuh ke Amazon Cognito. Tugas Anda adalah menentukan fitur dan sumber daya Amazon Cognito mana yang harus diakses pengguna layanan Anda. Kemudian, Anda harus mengirimkan permintaan kepada administrator IAM untuk mengubah izin pengguna layanan Anda. Tinjau informasi di halaman ini untuk memahami konsep Basic IAM. Untuk mempelajari lebih lanjut tentang bagaimana perusahaan Anda dapat menggunakan IAM dengan Amazon Cognito, lihat. [Cara kerja Amazon Cognito dengan IAM](#)

Administrator IAM - Jika Anda administrator IAM, Anda mungkin ingin mempelajari detail tentang cara menulis kebijakan untuk mengelola akses ke Amazon Cognito. Untuk melihat contoh kebijakan

berbasis identitas Amazon Cognito yang dapat Anda gunakan di IAM, lihat. [Contoh kebijakan berbasis identitas untuk Amazon Cognito](#)

## Mengautentikasi dengan identitas

Otentikasi adalah cara Anda masuk AWS menggunakan kredensial identitas Anda. Anda harus diautentikasi (masuk ke AWS) sebagai Pengguna root akun AWS, sebagai pengguna IAM, atau dengan mengasumsikan peran IAM.

Anda dapat masuk AWS sebagai identitas federasi dengan menggunakan kredensi yang disediakan melalui sumber identitas. AWS IAM Identity Center Pengguna (IAM Identity Center), autentikasi masuk tunggal perusahaan Anda, dan kredensi Google atau Facebook Anda adalah contoh identitas federasi. Saat Anda masuk sebagai identitas terfederasi, administrator Anda sebelumnya menyiapkan federasi identitas menggunakan peran IAM. Ketika Anda mengakses AWS dengan menggunakan federasi, Anda secara tidak langsung mengambil peran.

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

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 AWS alat, Anda harus menandatangani permintaan sendiri. Untuk informasi selengkapnya tentang penggunaan metode yang disarankan untuk menandatangani permintaan sendiri, lihat [Menandatangani permintaan AWS API](#) di Panduan Pengguna IAM.

Apa pun metode autentikasi yang digunakan, Anda mungkin diminta untuk menyediakan informasi keamanan tambahan. Misalnya, AWS merekomendasikan agar Anda menggunakan otentikasi multi-faktor (MFA) untuk meningkatkan keamanan akun Anda. Untuk mempelajari selengkapnya, lihat [Autentikasi multi-faktor](#) dalam Panduan Pengguna AWS IAM Identity Center dan [Menggunakan autentikasi multi-faktor \(MFA\) dalam AWS](#) dalam Panduan Pengguna IAM.

## Akun AWS pengguna root

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. 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](#) dalam Panduan Pengguna IAM.

## Identitas gabungan

Sebagai praktik terbaik, mewajibkan pengguna manusia, termasuk pengguna yang memerlukan akses administrator, untuk menggunakan federasi dengan penyedia identitas untuk mengakses Layanan AWS dengan menggunakan kredensi sementara.

Identitas federasi adalah pengguna dari direktori pengguna perusahaan Anda, penyedia identitas web, direktori Pusat Identitas AWS Directory Service, atau pengguna mana pun yang mengakses Layanan AWS dengan menggunakan kredensial yang disediakan melalui sumber identitas. Ketika identitas federasi mengakses Akun AWS, mereka mengambil peran, dan peran memberikan kredensi sementara.

Untuk manajemen akses terpusat, kami sarankan Anda menggunakan AWS IAM Identity Center. Anda dapat membuat pengguna dan grup di Pusat Identitas IAM, atau Anda dapat menghubungkan dan menyinkronkan ke sekumpulan pengguna dan grup di sumber identitas Anda sendiri untuk digunakan di semua aplikasi Akun AWS dan aplikasi Anda. Untuk informasi tentang Pusat Identitas IAM, lihat [Apakah itu Pusat Identitas IAM?](#) dalam Panduan Pengguna AWS IAM Identity Center .

## Pengguna dan grup IAM

[Pengguna IAM](#) adalah identitas dalam diri Anda Akun AWS yang memiliki izin khusus untuk satu orang atau aplikasi. Jika memungkinkan, kami merekomendasikan untuk mengandalkan kredensial sementara, bukan membuat pengguna IAM yang memiliki kredensial jangka panjang seperti kata sandi dan kunci akses. Namun, jika Anda memiliki kasus penggunaan tertentu yang memerlukan kredensial jangka panjang dengan pengguna IAM, kami merekomendasikan Anda merotasi kunci akses. Untuk informasi selengkapnya, lihat [Merotasi kunci akses secara teratur untuk kasus penggunaan yang memerlukan kredensial jangka panjang](#) dalam Panduan Pengguna IAM.

[Grup IAM](#) adalah identitas yang menentukan sekumpulan pengguna IAM. Anda tidak dapat masuk sebagai grup. Anda dapat menggunakan grup untuk menentukan izin bagi beberapa pengguna sekaligus. Grup mempermudah manajemen izin untuk sejumlah besar pengguna sekaligus. Misalnya, Anda dapat memiliki grup yang bernama IAMAdmins dan memberikan izin ke grup tersebut untuk mengelola sumber daya IAM.

Pengguna berbeda dari peran. Pengguna secara unik terkait dengan satu orang atau aplikasi, tetapi peran dimaksudkan untuk dapat digunakan oleh siapa pun yang membutuhkannya. Pengguna

memiliki kredensial jangka panjang permanen, tetapi peran memberikan kredensial sementara. Untuk mempelajari selengkapnya, lihat [Kapan harus membuat pengguna IAM \(bukan peran\)](#) dalam Panduan Pengguna IAM.

## Peran IAM

[Peran IAM](#) adalah identitas dalam diri Anda Akun AWS yang memiliki izin khusus. Peran 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 cara menggunakan peran, lihat [Menggunakan peran IAM](#) dalam Panduan Pengguna IAM.

Peran IAM dengan kredensial sementara berguna dalam situasi berikut:

- Akses pengguna terfederasi – Untuk menetapkan izin ke identitas terfederasi, Anda membuat peran dan menentukan izin untuk peran tersebut. Ketika identitas terfederasi mengautentikasi, identitas tersebut terhubung dengan peran dan diberi izin yang ditentukan oleh peran. Untuk informasi tentang peran untuk federasi, lihat [Membuat peran untuk Penyedia Identitas pihak ketiga](#) dalam Panduan Pengguna IAM. Jika menggunakan Pusat Identitas IAM, Anda harus mengonfigurasi set izin. Untuk mengontrol apa yang dapat diakses identitas Anda setelah identitas tersebut diautentikasi, Pusat Identitas IAM akan mengorelasikan set izin ke peran dalam IAM. Untuk informasi tentang set izin, lihat [Set 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 (prinsipal tepercaya) di akun lain untuk mengakses sumber daya di akun Anda. Peran adalah cara utama untuk memberikan akses lintas akun. Namun, dengan beberapa Layanan AWS, Anda dapat melampirkan kebijakan secara langsung ke sumber daya (alih-alih menggunakan peran sebagai proxy). Untuk mempelajari perbedaan antara peran dan kebijakan berbasis sumber daya untuk akses lintas akun, lihat [Bagaimana peran IAM berbeda dari kebijakan berbasis sumber daya](#) dalam Panduan Pengguna IAM.
- Akses lintas layanan — Beberapa Layanan AWS menggunakan fitur lain Layanan AWS. Sebagai contoh, ketika Anda memanggil 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. Ketika Anda menggunakan beberapa layanan, Anda mungkin melakukan sebuah tindakan yang kemudian menginisiasi 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 ketika mengajukan permintaan FAS, lihat [Sesi akses maju](#).
- Peran layanan – Peran layanan adalah [peran IAM](#) yang dijalankan 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 sebuah peran untuk mendelegasikan izin ke Layanan AWS](#) dalam Panduan pengguna IAM.
- Peran terkait layanan — Peran terkait layanan adalah jenis peran layanan yang ditautkan ke peran layanan. Layanan AWS Layanan tersebut dapat menjalankan 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.
- 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 dalam instans Amazon EC2](#) dalam Panduan Pengguna IAM.

Untuk mempelajari apakah kita harus menggunakan peran IAM atau pengguna IAM, lihat [Kapan harus membuat peran IAM \(bukan pengguna\)](#) dalam Panduan Pengguna IAM.

## Mengelola akses menggunakan kebijakan

Anda mengontrol akses AWS dengan membuat kebijakan dan melampirkannya ke AWS identitas atau sumber daya. Kebijakan adalah objek AWS yang, ketika dikaitkan dengan identitas atau sumber daya, menentukan izinnya. AWS mengevaluasi kebijakan ini ketika prinsipal (pengguna, pengguna root, atau sesi peran) membuat permintaan. Izin dalam kebijakan menentukan apakah permintaan

diizinkan atau ditolak. Sebagian besar kebijakan disimpan AWS sebagai dokumen JSON. Untuk informasi selengkapnya tentang struktur dan isi dokumen kebijakan JSON, lihat [Gambaran umum kebijakan JSON](#) dalam Panduan Pengguna IAM.

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Artinya, prinsipal manakah yang dapat melakukan tindakan pada sumber daya apa, dan dengan kondisi apa.

Secara default, pengguna dan peran tidak memiliki izin. Untuk memberikan izin kepada pengguna untuk melakukan tindakan di sumber daya yang mereka perlukan, administrator IAM dapat membuat kebijakan IAM. Administrator kemudian dapat menambahkan kebijakan IAM ke peran, dan pengguna dapat mengambil peran.

Kebijakan IAM mendefinisikan izin untuk suatu tindakan terlepas dari metode yang Anda gunakan untuk melakukan operasinya. Misalnya, anggaplah Anda memiliki kebijakan yang mengizinkan tindakan `iam:GetRole`. Pengguna dengan kebijakan tersebut bisa mendapatkan informasi peran dari AWS Management Console, API AWS CLI, atau AWS API.

## Kebijakan berbasis identitas

Kebijakan berbasis identitas adalah dokumen kebijakan izin JSON yang dapat Anda lampirkan ke sebuah identitas, seperti pengguna IAM, grup pengguna IAM, atau peran IAM. Kebijakan ini mengontrol jenis tindakan yang dapat dilakukan oleh pengguna dan peran, di sumber daya mana, dan berdasarkan kondisi seperti apa. Untuk mempelajari cara membuat kebijakan berbasis identitas, lihat [Membuat kebijakan IAM](#) dalam Panduan Pengguna IAM.

Kebijakan berbasis identitas dapat dikategorikan lebih lanjut sebagai kebijakan inline atau kebijakan yang dikelola. Kebijakan inline disematkan langsung ke satu pengguna, grup, atau peran. Kebijakan terkelola adalah kebijakan mandiri yang dapat Anda lampirkan ke beberapa pengguna, grup, dan peran dalam Akun AWS. Kebijakan AWS terkelola mencakup kebijakan terkelola dan kebijakan yang dikelola pelanggan. Untuk mempelajari cara memilih antara kebijakan yang dikelola atau kebijakan inline, lihat [Memilih antara kebijakan yang dikelola dan kebijakan inline](#) dalam Panduan Pengguna IAM.

## Kebijakan berbasis sumber daya

Kebijakan berbasis sumber daya adalah dokumen kebijakan JSON yang Anda lampirkan ke sumber daya. Contoh kebijakan berbasis sumber daya adalah kebijakan kepercayaan peran IAM dan kebijakan bucket Amazon S3. Dalam layanan yang mendukung kebijakan berbasis sumber daya,



administrator layanan dapat menggunakannya untuk mengontrol akses ke sumber daya tertentu. Untuk sumber daya tempat kebijakan dilampirkan, kebijakan menentukan tindakan apa yang dapat dilakukan oleh prinsipal tertentu pada sumber daya tersebut dan dalam kondisi apa. Anda harus [menentukan prinsipal](#) dalam kebijakan berbasis sumber daya. Prinsipal dapat mencakup akun, pengguna, peran, pengguna federasi, atau. Layanan AWS

Kebijakan berbasis sumber daya merupakan kebijakan inline yang terletak di layanan tersebut. Anda tidak dapat menggunakan kebijakan AWS terkelola dari IAM dalam kebijakan berbasis sumber daya.

## Daftar kontrol akses (ACL)

Daftar kontrol akses (ACL) mengendalikan prinsipal mana (anggota akun, pengguna, atau peran) yang memiliki izin untuk mengakses sumber daya. ACL serupa dengan kebijakan berbasis sumber daya, meskipun kebijakan tersebut tidak menggunakan format dokumen kebijakan JSON.

Amazon S3, AWS WAF, dan Amazon VPC adalah contoh layanan yang mendukung ACL. Untuk mempelajari ACL selengkapnya, lihat [Gambaran umum daftar kontrol akses \(ACL\)](#) dalam Panduan Developer Amazon Simple Storage Service.

## Jenis-jenis kebijakan lain

AWS mendukung jenis kebijakan tambahan yang kurang umum. Jenis-jenis kebijakan ini dapat mengatur izin maksimum yang diberikan kepada Anda oleh jenis kebijakan yang lebih umum.

- Batasan izin – Batasan izin adalah fitur lanjutan tempat Anda mengatur izin maksimum yang dapat diberikan oleh kebijakan berbasis identitas ke entitas IAM (pengguna IAM atau peran IAM). Anda dapat menetapkan batasan izin untuk suatu entitas. Izin yang dihasilkan adalah perpotongan antara kebijakan berbasis identitas milik entitas dan batasan izinnya. Kebijakan berbasis sumber daya yang menentukan pengguna atau peran dalam bidang `Principal` tidak dibatasi oleh batasan izin. Penolakan eksplisit dalam salah satu kebijakan ini akan menggantikan pemberian izin. Untuk informasi selengkapnya tentang batasan izin, lihat [Batasan izin untuk entitas IAM](#) dalam Panduan Pengguna IAM.
- Kebijakan kontrol layanan (SCP) — SCP adalah kebijakan JSON yang menentukan izin maksimum untuk organisasi atau unit organisasi (OU) di. AWS Organizations AWS Organizations adalah layanan untuk mengelompokkan dan mengelola secara terpusat beberapa Akun AWS yang dimiliki bisnis Anda. Jika Anda mengaktifkan semua fitur di organisasi, Anda dapat menerapkan kebijakan kontrol layanan (SCP) ke salah satu atau semua akun Anda. SCP membatasi izin untuk entitas di akun anggota, termasuk masing-masing. Pengguna root akun AWS Untuk informasi selengkapnya tentang Organisasi dan SCP, lihat [Cara kerja SCP](#) dalam Panduan Pengguna AWS Organizations .



- Kebijakan sesi – Kebijakan sesi adalah kebijakan lanjutan yang Anda berikan sebagai parameter ketika Anda membuat sesi sementara secara programatis untuk peran atau pengguna terfederasi. Izin sesi yang dihasilkan adalah perpotongan antara kebijakan berbasis identitas pengguna atau peran dan kebijakan sesi. Izin juga bisa datang dari kebijakan berbasis sumber daya. Penolakan eksplisit dalam salah satu kebijakan ini akan menggantikan pemberian izin. Untuk informasi selengkapnya, lihat [Kebijakan sesi](#) dalam Panduan Pengguna IAM.

## Berbagai jenis kebijakan

Ketika beberapa jenis kebijakan berlaku pada suatu permintaan, izin yang dihasilkan lebih rumit untuk dipahami. Untuk mempelajari cara AWS menentukan apakah akan mengizinkan permintaan saat beberapa jenis kebijakan terlibat, lihat [Logika evaluasi kebijakan](#) di Panduan Pengguna IAM.

## Cara kerja Amazon Cognito dengan IAM

Sebelum Anda menggunakan IAM untuk mengelola akses ke Amazon Cognito, pelajari fitur IAM apa yang tersedia untuk digunakan dengan Amazon Cognito.

Fitur IAM yang dapat Anda gunakan dengan Amazon Cognito

Fitur IAM	Dukungan Amazon Cognito
<a href="#">Kebijakan berbasis identitas</a>	Ya
<a href="#">Kebijakan berbasis sumber daya</a>	Tidak
<a href="#">Tindakan kebijakan</a>	Ya
<a href="#">Sumber daya kebijakan</a>	Ya
<a href="#">Kunci kondisi kebijakan</a>	Ya
<a href="#">ACL</a>	Tidak
<a href="#">ABAC (tanda dalam kebijakan)</a>	Parsial
<a href="#">Kredensial sementara</a>	Ya
<a href="#">Izin prinsipal</a>	Tidak

Fitur IAM	Dukungan Amazon Cognito
<a href="#">Peran layanan</a>	Ya
<a href="#">Peran terkait layanan</a>	Ya

Untuk mendapatkan tampilan tingkat tinggi tentang cara kerja Amazon Cognito dan layanan AWS lainnya dengan sebagian besar fitur IAM, [AWS lihat layanan yang bekerja dengan IAM di Panduan Pengguna IAM](#).

## Kebijakan berbasis identitas untuk Amazon Cognito

Mendukung kebijakan berbasis identitas	Ya
----------------------------------------	----

Kebijakan berbasis identitas adalah dokumen kebijakan izin JSON yang dapat Anda lampirkan ke sebuah identitas, seperti pengguna IAM, grup pengguna IAM, atau peran IAM. Kebijakan ini mengontrol jenis tindakan yang dapat dilakukan oleh pengguna dan peran, di sumber daya mana, dan berdasarkan kondisi seperti apa. Untuk mempelajari cara membuat kebijakan berbasis identitas, lihat [Membuat kebijakan IAM](#) dalam Panduan Pengguna IAM.

Dengan kebijakan berbasis identitas IAM, Anda dapat menentukan secara spesifik apakah tindakan dan sumber daya diizinkan atau ditolak, serta kondisi yang menjadi dasar dikabulkan atau ditolaknya tindakan tersebut. Anda tidak dapat menentukan secara spesifik prinsipal dalam sebuah kebijakan berbasis identitas karena prinsipal berlaku bagi pengguna atau peran yang melekat kepadanya. Untuk mempelajari semua elemen yang dapat Anda gunakan dalam kebijakan JSON, lihat [Referensi elemen kebijakan JSON IAM](#) dalam Panduan Pengguna IAM.

## Contoh kebijakan berbasis identitas untuk Amazon Cognito

Untuk melihat contoh kebijakan berbasis identitas Amazon Cognito, lihat. [Contoh kebijakan berbasis identitas untuk Amazon Cognito](#)

## Kebijakan berbasis sumber daya dalam Amazon Cognito

Mendukung kebijakan berbasis sumber daya	Tidak
------------------------------------------	-------

Kebijakan berbasis sumber daya adalah dokumen kebijakan JSON yang Anda lampirkan ke sumber daya. Contoh kebijakan berbasis sumber daya adalah kebijakan kepercayaan peran IAM dan kebijakan bucket Amazon S3. Dalam layanan yang mendukung kebijakan berbasis sumber daya, administrator layanan dapat menggunakannya untuk mengontrol akses ke sumber daya tertentu. Untuk sumber daya tempat kebijakan dilampirkan, kebijakan menentukan tindakan apa yang dapat dilakukan oleh prinsipal tertentu pada sumber daya tersebut dan dalam kondisi apa. Anda harus [menentukan prinsipal](#) dalam kebijakan berbasis sumber daya. Prinsipal dapat mencakup akun, pengguna, peran, pengguna federasi, atau. Layanan AWS

Untuk mengaktifkan akses lintas akun, Anda dapat menentukan secara spesifik seluruh akun atau entitas IAM di akun lain sebagai prinsipal dalam kebijakan berbasis sumber daya. Menambahkan prinsipal akun silang ke kebijakan berbasis sumber daya hanya setengah dari membangun hubungan kepercayaan. Ketika prinsipal dan sumber daya berbeda Akun AWS, administrator IAM di akun tepercaya juga harus memberikan izin entitas utama (pengguna atau peran) untuk mengakses sumber daya. Mereka memberikan izin dengan melampirkan kebijakan berbasis identitas kepada entitas. Namun, jika kebijakan berbasis sumber daya memberikan akses ke prinsipal dalam akun yang sama, tidak diperlukan kebijakan berbasis identitas tambahan. Untuk informasi selengkapnya, lihat [Bagaimana peran IAM berbeda dari kebijakan berbasis sumber daya](#) dalam Panduan Pengguna IAM.

## Tindakan kebijakan untuk Amazon Cognito

Mendukung tindakan kebijakan

Ya

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Artinya, prinsipal manakah yang dapat melakukan tindakan pada sumber daya apa, dan dengan kondisi apa.

Elemen `Action` dari kebijakan JSON menjelaskan tindakan yang dapat Anda gunakan untuk mengizinkan atau menolak akses dalam sebuah kebijakan. Tindakan kebijakan biasanya memiliki nama yang sama dengan operasi AWS API terkait. Ada beberapa pengecualian, misalnya tindakan hanya izin yang tidak memiliki operasi API yang cocok. Ada juga beberapa operasi yang memerlukan beberapa tindakan dalam suatu kebijakan. Tindakan tambahan ini disebut tindakan dependen.

Menyertakan tindakan dalam kebijakan untuk memberikan izin untuk melakukan operasi terkait.

Untuk melihat daftar tindakan Amazon Cognito, lihat [Tindakan yang ditentukan oleh Amazon Cognito](#) di Referensi Otorisasi Layanan.

Tindakan kebijakan di Amazon Cognito menggunakan awalan berikut sebelum tindakan:

```
cognito-identity
```

Untuk menetapkan secara spesifik beberapa tindakan dalam satu pernyataan, pisahkan tindakan tersebut dengan koma.

```
"Action": [  
  "cognito-identity:action1",  
  "cognito-identity:action2"  
]
```

## API yang ditandatangani versus tidak ditandatangani

Saat menandatangani permintaan Amazon Cognito API dengan AWS kredensial, Anda dapat membatasinya dalam kebijakan (IAM). AWS Identity and Access Management Permintaan API yang harus Anda tandatangi dengan AWS kredensial mencakup login di sisi serverAdminInitiateAuth, dan tindakan yang membuat, melihat, atau memodifikasi resource Amazon Cognito Anda seperti. UpdateUserPool Untuk informasi selengkapnya tentang permintaan API yang ditandatangani, lihat [Menandatangani permintaan AWS API](#).

Karena Amazon Cognito adalah produk identitas konsumen untuk aplikasi yang ingin Anda sediakan untuk umum, Anda memiliki akses ke API yang tidak ditandatangani berikut. Aplikasi Anda membuat permintaan API ini untuk pengguna dan calon pengguna Anda. Beberapa API tidak memerlukan otorisasi sebelumnya, seperti InitiateAuth memulai sesi otentikasi baru. Beberapa API menggunakan token akses atau kunci sesi untuk otorisasi, seperti menyelesaikan penyiapan MFA VerifySoftwareToken untuk pengguna yang memiliki sesi otentikasi yang sudah ada. API kumpulan pengguna Amazon Cognito resmi yang tidak ditandatangani mendukung Session parameter AccessToken atau dalam sintaks permintaan seperti yang ditampilkan di Referensi API Amazon [Cognito](#). API Identitas Amazon Cognito yang tidak ditandatangani mendukung parameter seperti IdentityId yang ditampilkan di Referensi API Identitas [Federasi Amazon Cognito](#).

Untuk informasi selengkapnya tentang model otorisasi dan peran operasi API kumpulan pengguna Amazon Cognito, lihat. [Kumpulan pengguna Amazon Cognito operasi API yang diautentikasi dan tidak diautentikasi](#)

## Operasi API kumpulan identitas Amazon Cognito

- `GetId`
- `GetOpenIdToken`
- `GetCredentialsForIdentity`
- `UnlinkIdentity`

## Operasi API kumpulan pengguna Amazon Cognito

- `AssociateSoftwareToken`
- `ChangePassword`
- `ConfirmDevice`
- `ConfirmForgotPassword`
- `ConfirmSignUp`
- `DeleteUser`
- `DeleteUserAttributes`
- `ForgetDevice`
- `ForgotPassword`
- `GetDevice`
- `GetUser`
- `GetUserAttributeVerificationCode`
- `GlobalSignOut`
- `InitiateAuth`
- `ListDevices`
- `ResendConfirmationCode`
- `RespondToAuthChallenge`
- `RevokeToken`
- `SetUserMFAPreference`
- `SetUserSettings`
- `SignUp`
- `UpdateAuthEventFeedback`
- `UpdateDeviceStatus`

- UpdateUserAttributes
- VerifySoftwareToken
- VerifyUserAttribute

Untuk melihat contoh kebijakan berbasis identitas Amazon Cognito, lihat. [Contoh kebijakan berbasis identitas untuk Amazon Cognito](#)

## Sumber daya kebijakan untuk Amazon Cognito

Mendukung sumber daya kebijakan	Ya
---------------------------------	----

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Artinya, prinsipal manakah yang dapat melakukan tindakan pada sumber daya apa, dan dengan kondisi apa.

Elemen kebijakan JSON Resource menentukan objek yang menjadi target penerapan tindakan. Pernyataan harus menyertakan elemen Resource atau NotResource. Praktik terbaiknya, tentukan sumber daya menggunakan [Amazon Resource Name \(ARN\)](#). Anda dapat melakukan ini untuk tindakan yang mendukung jenis sumber daya tertentu, yang dikenal sebagai izin tingkat sumber daya.

Untuk tindakan yang tidak support izin tingkat sumber daya, seperti operasi daftar, gunakan wildcard (\*) untuk menunjukkan bahwa pernyataan berlaku untuk semua sumber daya.

```
"Resource": "*" 
```

Nama sumber daya Amazon (ARN)

ARN untuk identitas federasi Amazon Cognito

Di kolam identitas Amazon Cognito (identitas gabungan), dimungkinkan untuk membatasi akses pengguna IAM ke kolam identitas tertentu, menggunakan format Amazon Resource Name (ARN), seperti dalam contoh berikut. Untuk informasi selengkapnya tentang ARN, lihat pengidentifikasi [IAM](#).

```
arn:aws:cognito-identity:REGION:ACCOUNT_ID:identitypool/IDENTITY_POOL_ID
```

## ARN untuk Sinkronisasi Amazon Cognito

Di Amazon Cognito Sync, pelanggan juga dapat membatasi akses dengan ID kolam identitas, ID identitas, dan nama set data.

Untuk API yang beroperasi pada kolam identitas, format ARN kolam identitas sama dengan untuk Identitas Federasi Amazon Cognito, kecuali bahwa nama layanannya adalah `cognito-sync` bukan `cognito-identity`:

```
arn:aws:cognito-sync:REGION:ACCOUNT_ID:identitypool/IDENTITY_POOL_ID
```

Untuk API yang beroperasi pada identitas tunggal, seperti `RegisterDevice`, Anda dapat merujuk ke identitas individu dengan format ARN berikut:

```
arn:aws:cognito-sync:REGION:ACCOUNT_ID:identitypool/IDENTITY_POOL_ID/identity/IDENTITY_ID
```

Untuk API yang beroperasi pada set data, seperti `UpdateRecords`, dan `ListRecords`, Anda dapat merujuk ke set data dengan format ARN berikut:

```
arn:aws:cognito-sync:REGION:ACCOUNT_ID:identitypool/IDENTITY_POOL_ID/identity/IDENTITY_ID/dataset/DATASET_NAME
```

## ARN untuk kumpulan pengguna Amazon Cognito

Untuk Amazon Cognito Your User Pools, dimungkinkan untuk membatasi akses pengguna ke kumpulan pengguna tertentu, menggunakan format ARN berikut:

```
arn:aws:cognito-idp:REGION:ACCOUNT_ID:userpool/USER_POOL_ID
```

Untuk melihat daftar jenis sumber daya Amazon Cognito dan ARNnya, lihat Sumber daya yang [ditetapkan oleh Amazon Cognito](#) dalam Referensi Otorisasi Layanan. Untuk mempelajari tindakan mana yang dapat menentukan ARN dari setiap sumber daya, lihat [Tindakan yang ditentukan oleh Amazon Cognito](#).

Untuk melihat contoh kebijakan berbasis identitas Amazon Cognito, lihat. [Contoh kebijakan berbasis identitas untuk Amazon Cognito](#)

## Kunci kondisi kebijakan untuk Amazon Cognito

Mendukung kunci kondisi kebijakan khusus layanan	Ya
--------------------------------------------------	----

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Artinya, prinsipal manakah yang dapat melakukan tindakan pada sumber daya apa, dan dengan kondisi apa.

Elemen `Condition` (atau blok `Condition`) akan memungkinkan Anda menentukan kondisi yang menjadi dasar suatu pernyataan berlaku. Elemen `Condition` bersifat opsional. Anda dapat membuat ekspresi bersyarat yang menggunakan [operator kondisi](#), misalnya sama dengan atau kurang dari, untuk mencocokkan kondisi dalam kebijakan dengan nilai-nilai yang diminta.

Jika Anda menentukan beberapa elemen `Condition` dalam sebuah pernyataan, atau beberapa kunci dalam elemen `Condition` tunggal, maka AWS akan mengevaluasinya menggunakan operasi AND logis. Jika Anda menentukan beberapa nilai untuk satu kunci kondisi, AWS mengevaluasi kondisi menggunakan OR operasi logis. Semua kondisi harus dipenuhi sebelum izin pernyataan diberikan.

Anda juga dapat menggunakan variabel placeholder saat menentukan kondisi. Sebagai contoh, Anda dapat memberikan izin kepada pengguna IAM untuk mengakses sumber daya hanya jika izin tersebut mempunyai tag yang sesuai dengan nama pengguna IAM mereka. Untuk informasi selengkapnya, lihat [Elemen kebijakan IAM: variabel dan tag](#) dalam Panduan Pengguna IAM.

AWS mendukung kunci kondisi global dan kunci kondisi khusus layanan. Untuk melihat semua kunci kondisi AWS global, lihat [kunci konteks kondisi AWS global](#) di Panduan Pengguna IAM.

Untuk melihat daftar kunci kondisi Amazon Cognito, lihat Kunci kondisi [untuk Amazon Cognito](#) di Referensi Otorisasi Layanan. Untuk mempelajari dengan tindakan dan sumber daya mana Anda dapat menggunakan kunci syarat, lihat [Tindakan yang ditentukan oleh Amazon Cognito](#).

Untuk melihat contoh kebijakan berbasis identitas Amazon Cognito, lihat. [Contoh kebijakan berbasis identitas untuk Amazon Cognito](#)

## Daftar kontrol akses (ACL) di Amazon Cognito

Mendukung ACL	Tidak
---------------	-------



Daftar kontrol akses (ACL) mengendalikan pengguna utama mana (anggota akun, pengguna, atau peran) yang memiliki izin untuk mengakses sumber daya. ACL serupa dengan kebijakan berbasis sumber daya, meskipun kebijakan tersebut tidak menggunakan format dokumen kebijakan JSON.

## Kontrol akses berbasis atribut (ABAC) dengan Amazon Cognito

Mendukung ABAC (tanda dalam kebijakan) Parsial

Kontrol akses berbasis atribut (ABAC) adalah strategi otorisasi yang menentukan izin berdasarkan atribut. Dalam AWS, atribut ini disebut tag. Anda dapat melampirkan tag ke entitas IAM (pengguna atau peran) dan ke banyak AWS sumber daya. Penandaan ke entitas dan sumber daya adalah langkah pertama dari ABAC. Kemudian rancanglah kebijakan ABAC untuk mengizinkan operasi ketika tag milik prinsipal cocok dengan tag yang ada di sumber daya yang ingin diakses.

ABAC sangat berguna di lingkungan yang berkembang dengan cepat dan berguna di situasi saat manajemen kebijakan menjadi rumit.

Untuk mengendalikan akses berdasarkan tag, berikan informasi tentang tag di [elemen kondisi](#) dari kebijakan menggunakan kunci kondisi `aws:ResourceTag/key-name`, `aws:RequestTag/key-name`, atau `aws:TagKeys`.

Jika sebuah layanan mendukung ketiga kunci kondisi untuk setiap jenis sumber daya, nilainya adalah Ya untuk layanan tersebut. Jika suatu layanan mendukung ketiga kunci kondisi untuk hanya beberapa jenis sumber daya, nilainya adalah Parsial.

Untuk informasi selengkapnya tentang ABAC, lihat [Apa itu ABAC?](#) dalam Panduan Pengguna IAM. Untuk melihat tutorial yang menguraikan langkah-langkah pengaturan ABAC, lihat [Menggunakan kontrol akses berbasis atribut \(ABAC\)](#) dalam Panduan Pengguna IAM.

## Menggunakan kredensial sementara dengan Amazon Cognito

Mendukung penggunaan kredensial sementara Ya

Beberapa Layanan AWS tidak berfungsi saat Anda masuk menggunakan kredensial sementara. Untuk informasi tambahan, termasuk yang Layanan AWS bekerja dengan kredensial sementara, lihat [Layanan AWS yang bekerja dengan IAM di Panduan Pengguna IAM](#).

Anda menggunakan kredensial sementara jika Anda masuk AWS Management Console menggunakan metode apa pun kecuali nama pengguna dan kata sandi. Misalnya, ketika Anda mengakses AWS menggunakan tautan masuk tunggal (SSO) perusahaan Anda, proses tersebut secara otomatis membuat kredensial sementara. Anda juga akan secara otomatis membuat kredensial sementara ketika Anda masuk ke konsol sebagai seorang pengguna lalu beralih peran. Untuk informasi selengkapnya tentang peralihan peran, lihat [Peralihan peran \(konsol\)](#) dalam Panduan Pengguna IAM.

Anda dapat membuat kredensial sementara secara manual menggunakan API AWS CLI atau AWS . Anda kemudian dapat menggunakan kredensial sementara tersebut untuk mengakses AWS . AWS merekomendasikan agar Anda menghasilkan kredensial sementara secara dinamis alih-alih menggunakan kunci akses jangka panjang. Untuk informasi selengkapnya, lihat [Kredensial keamanan sementara di IAM](#).

## Izin utama lintas layanan untuk Amazon Cognito

Mendukung sesi akses maju (FAS)

Tidak

Saat Anda menggunakan pengguna atau peran IAM untuk melakukan tindakan AWS, Anda dianggap sebagai prinsipal. Ketika Anda menggunakan beberapa layanan, Anda mungkin melakukan sebuah tindakan yang kemudian menginisiasi 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 ketika mengajukan permintaan FAS, lihat [Sesi akses maju](#).

## Peran layanan untuk Amazon Cognito

Mendukung peran layanan

Ya

Peran layanan adalah [peran IAM](#) yang diambil oleh sebuah layanan untuk melakukan tindakan atas nama Anda. Administrator IAM dapat membuat, mengubah, dan menghapus peran layanan dari dalam IAM. Untuk informasi selengkapnya, lihat [Membuat sebuah peran untuk mendelegasikan izin ke Layanan AWS](#) dalam Panduan pengguna IAM.

Untuk detail tentang peran layanan Amazon Cognito, lihat [Aktifkan sinkronisasi push](#) dan [Sinkronisasi push](#)

#### Warning

Mengubah izin untuk peran layanan dapat merusak fungsionalitas Amazon Cognito. Edit peran layanan hanya jika Amazon Cognito memberikan panduan untuk melakukannya.

## Peran terkait layanan untuk Amazon Cognito

Mendukung peran terkait layanan	Ya
---------------------------------	----

Peran terkait layanan adalah jenis peran layanan yang ditautkan ke. Layanan AWS Layanan tersebut dapat menjalankan 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.

Untuk detail tentang membuat atau mengelola peran terkait layanan Amazon Cognito, lihat [Menggunakan peran terkait layanan untuk Amazon Cognito](#)

## Contoh kebijakan berbasis identitas untuk Amazon Cognito

Secara default, pengguna dan peran tidak memiliki izin untuk membuat atau memodifikasi sumber daya Amazon Cognito. Mereka juga tidak dapat melakukan tugas dengan menggunakan AWS Management Console, AWS Command Line Interface (AWS CLI), atau AWS API. Untuk memberikan izin kepada pengguna untuk melakukan tindakan di sumber daya yang mereka perlukan, administrator IAM dapat membuat kebijakan IAM. Administrator kemudian akan dapat menambahkan kebijakan IAM ke peran, dan pengguna dapat mengambil peran.

Untuk mempelajari cara membuat kebijakan berbasis identitas IAM menggunakan contoh dokumen kebijakan JSON ini, lihat [Membuat kebijakan IAM](#) dalam Panduan Pengguna IAM.

Untuk detail tentang tindakan dan jenis sumber daya yang ditentukan oleh Amazon Cognito, termasuk format ARN untuk setiap jenis sumber daya, lihat [Kunci tindakan, sumber daya, dan kondisi untuk Amazon Cognito](#) di Referensi Otorisasi Layanan.

### Topik

- [Praktik terbaik kebijakan](#)
- [Menggunakan konsol Amazon Cognito](#)
- [Mengizinkan pengguna melihat izin mereka sendiri](#)
- [Membatasi akses konsol ke kumpulan identitas tertentu](#)
- [Mengizinkan akses ke kumpulan data tertentu untuk semua identitas dalam kumpulan](#)

## Praktik terbaik kebijakan

Kebijakan berbasis identitas menentukan apakah seseorang dapat membuat, mengakses, atau menghapus sumber daya Amazon Cognito di akun Anda. Tindakan ini membuat Akun AWS Anda dikenai biaya. Ketika Anda membuat atau mengedit kebijakan berbasis identitas, ikuti panduan dan rekomendasi ini:

- Mulailah dengan 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 Anda Akun AWS. Kami menyarankan Anda mengurangi izin lebih lanjut dengan menentukan kebijakan yang dikelola AWS pelanggan yang khusus untuk kasus penggunaan Anda. Untuk informasi selengkapnya, lihat [Kebijakan yang dikelola AWS](#) atau [Kebijakan yang dikelola AWS untuk fungsi tugas](#) dalam Panduan Pengguna IAM.
- Menerapkan izin dengan hak akses paling rendah – Ketika Anda menetapkan izin dengan kebijakan IAM, hanya berikan izin yang diperlukan untuk melakukan tugas. Anda melakukannya dengan mendefinisikan tindakan yang dapat diambil pada sumber daya tertentu dalam kondisi tertentu, yang juga dikenal sebagai izin dengan hak akses paling rendah. Untuk informasi selengkapnya tentang cara menggunakan IAM untuk mengajukan izin, lihat [Kebijakan dan izin dalam IAM](#) dalam Panduan Pengguna IAM.
- Gunakan kondisi dalam kebijakan IAM untuk membatasi akses lebih lanjut – Anda dapat menambahkan suatu kondisi ke kebijakan Anda untuk membatasi akses ke tindakan dan sumber daya. Sebagai contoh, 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 jika digunakan melalui yang spesifik Layanan AWS, seperti AWS CloudFormation. Untuk informasi selengkapnya, lihat [Elemen kebijakan JSON IAM: Kondisi](#) dalam Panduan Pengguna IAM.
- Gunakan IAM Access Analyzer untuk memvalidasi kebijakan IAM Anda untuk memastikan izin yang aman dan fungsional – IAM Access Analyzer memvalidasi kebijakan baru dan yang sudah

ada sehingga kebijakan tersebut mematuhi bahasa kebijakan IAM (JSON) dan praktik terbaik IAM. IAM Access Analyzer menyediakan lebih dari 100 pemeriksaan kebijakan dan rekomendasi yang dapat ditindaklanjuti untuk membantu Anda membuat kebijakan yang aman dan fungsional. Untuk informasi selengkapnya, lihat [Validasi kebijakan IAM Access Analyzer](#) dalam Panduan Pengguna IAM.

- Memerlukan otentikasi multi-faktor (MFA) - Jika Anda memiliki skenario yang mengharuskan pengguna IAM atau pengguna root di Anda, Akun AWS aktifkan MFA untuk keamanan tambahan. Untuk meminta MFA ketika operasi API dipanggil, tambahkan kondisi MFA pada kebijakan Anda. Untuk informasi selengkapnya, lihat [Mengonfigurasi akses API yang dilindungi MFA](#) dalam Panduan Pengguna IAM.

Untuk informasi selengkapnya tentang praktik terbaik dalam IAM, lihat [Praktik terbaik keamanan dalam IAM](#) dalam Panduan Pengguna IAM.

#### Note

Versi asli dan baru dari konsol Amazon Cognito memiliki perilaku mendasar yang berbeda saat Anda melihat dan memodifikasi sumber daya Amazon Cognito Anda. Jika Anda memberikan izin untuk tindakan di bawah awalan `cognito-idp` layanan hanya jika `aws:ViaAWSService` kondisinya benar, prinsipal IAM yang terpengaruh mungkin efektif untuk sumber daya Amazon Cognito di konsol asli, tetapi bukan konsol baru. Untuk bekerja di konsol Amazon Cognito, jangan setel `aws:ViaAWSService` kondisi izin Amazon Cognito dalam kebijakan IAM Anda.

## Menggunakan konsol Amazon Cognito

Untuk mengakses konsol Amazon Cognito, Anda harus memiliki rangkaian izin minimum. Izin ini harus memungkinkan Anda untuk membuat daftar dan melihat detail tentang sumber daya Amazon Cognito di Anda. Akun AWS Jika Anda membuat kebijakan berbasis identitas yang lebih ketat daripada izin minimum yang diperlukan, konsol tidak akan berfungsi sebagaimana mestinya untuk entitas (pengguna atau peran) dengan kebijakan tersebut.

Anda tidak perlu mengizinkan izin konsol minimum untuk pengguna yang melakukan panggilan hanya ke AWS CLI atau AWS API. Sebagai gantinya, izinkan akses hanya ke tindakan yang sesuai dengan operasi API yang coba mereka lakukan.

Untuk memastikan bahwa pengguna dan peran masih dapat menggunakan konsol Amazon Cognito, lampirkan juga `Amazon ConsoleAccess Cognito ReadOnly` AWS atau kebijakan terkelola ke entitas. Untuk informasi selengkapnya, lihat [Menambah izin untuk pengguna](#) dalam Panduan Pengguna IAM.

## Mengizinkan pengguna melihat izin mereka sendiri

Contoh ini menunjukkan cara membuat kebijakan yang mengizinkan pengguna IAM melihat kebijakan inline dan terkelola yang dilampirkan ke identitas pengguna mereka. Kebijakan ini mencakup izin untuk menyelesaikan tindakan ini di konsol atau menggunakan API atau secara terprogram. `AWS CLI` `AWS`

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ViewOwnUserInfo",
      "Effect": "Allow",
      "Action": [
        "iam:GetUserPolicy",
        "iam:ListGroupsWithUser",
        "iam:ListAttachedUserPolicies",
        "iam:ListUserPolicies",
        "iam:GetUser"
      ],
      "Resource": ["arn:aws:iam::*:user/${aws:username}"]
    },
    {
      "Sid": "NavigateInConsole",
      "Effect": "Allow",
      "Action": [
        "iam:GetGroupPolicy",
        "iam:GetPolicyVersion",
        "iam:GetPolicy",
        "iam:ListAttachedGroupPolicies",
        "iam:ListGroupPolicies",
        "iam:ListPolicyVersions",
        "iam:ListPolicies",
        "iam:ListUsers"
      ],
      "Resource": "*"
    }
  ]
}
```

```
]
}
```

## Membatasi akses konsol ke kumpulan identitas tertentu

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cognito-identity:ListIdentityPools"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "cognito-identity:*"
      ],
      "Resource": "arn:aws:cognito-identity:us-east-1:0123456789:identitypool/us-east-1:1a1a1a1a-ffff-1111-9999-12345678"
    },
    {
      "Effect": "Allow",
      "Action": [
        "cognito-sync:*"
      ],
      "Resource": "arn:aws:cognito-sync:us-east-1:0123456789:identitypool/us-east-1:1a1a1a1a-ffff-1111-9999-12345678"
    }
  ]
}
```

## Mengizinkan akses ke kumpulan data tertentu untuk semua identitas dalam kumpulan

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

```
        "cognito-sync:ListRecords",
        "cognito-sync:UpdateRecords"
    ],
    "Resource": "arn:aws:cognito-sync:us-east-1:0123456789:identitypool/us-
east-1:1a1a1a1a-ffff-1111-9999-12345678/identity/*/dataset/UserProfile"
    }
]
}
```

## Pemecahan masalah identitas dan akses Amazon Cognito

Gunakan informasi berikut untuk membantu Anda mendiagnosis dan memperbaiki masalah umum yang mungkin Anda temui saat bekerja dengan Amazon Cognito dan IAM.

### Topik

- [Saya tidak berwenang untuk melakukan tindakan di Amazon Cognito](#)
- [Saya tidak berwenang untuk melakukan iam: PassRole](#)
- [Saya seorang administrator dan ingin mengizinkan orang lain mengakses Amazon Cognito](#)
- [Saya ingin mengizinkan orang di luar AWS akun saya untuk mengakses sumber daya Amazon Cognito saya](#)

### Saya tidak berwenang untuk melakukan tindakan di Amazon Cognito

Jika Anda menerima pesan kesalahan bahwa Anda tidak memiliki otorisasi untuk melakukan tindakan, kebijakan Anda harus diperbarui agar Anda dapat melakukan tindakan tersebut.

Contoh kesalahan berikut terjadi ketika pengguna IAM mateojackson mencoba menggunakan konsol untuk melihat detail tentang suatu sumber daya *my-example-widget* rekaan, tetapi tidak memiliki izin `cognito-identity:GetWidget` rekaan.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
cognito-identity:GetWidget on resource: my-example-widget
```

Dalam hal ini, kebijakan untuk pengguna mateojackson harus diperbarui untuk mengizinkan akses ke sumber daya *my-example-widget* dengan menggunakan tindakan `cognito-identity:GetWidget`.



Jika Anda memerlukan bantuan, hubungi AWS administrator Anda. Administrator Anda adalah orang yang memberi Anda kredensial masuk.

## Saya tidak berwenang untuk melakukan iam: PassRole

Jika Anda menerima kesalahan bahwa Anda tidak diizinkan untuk melakukan `iam:PassRole` tindakan, kebijakan Anda harus diperbarui agar Anda dapat meneruskan peran ke Amazon Cognito.

Beberapa Layanan AWS memungkinkan Anda untuk meneruskan peran yang ada ke layanan tersebut alih-alih membuat peran layanan baru atau peran terkait layanan. Untuk melakukannya, Anda harus memiliki izin untuk meneruskan peran ke layanan.

Contoh kesalahan berikut terjadi ketika pengguna IAM bernama `marymajor` mencoba menggunakan konsol untuk melakukan tindakan di Amazon Cognito. Namun, tindakan tersebut memerlukan layanan untuk mendapatkan izin yang diberikan oleh peran layanan. Mary tidak memiliki izin untuk meneruskan peran tersebut pada layanan.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
iam:PassRole
```

Dalam kasus ini, kebijakan Mary harus diperbarui agar dia mendapatkan izin untuk melakukan tindakan `iam:PassRole` tersebut.

Jika Anda memerlukan bantuan, hubungi AWS administrator Anda. Administrator Anda adalah orang yang memberi Anda kredensial masuk.

## Saya seorang administrator dan ingin mengizinkan orang lain mengakses Amazon Cognito

Untuk mengizinkan orang lain mengakses Amazon Cognito, Anda harus membuat entitas IAM (pengguna atau peran) untuk orang atau aplikasi yang memerlukan akses. Mereka akan menggunakan kredensial untuk entitas tersebut untuk mengakses AWS. Anda kemudian harus melampirkan kebijakan ke entitas yang memberi mereka izin yang benar di Amazon Cognito.

Untuk segera mulai, lihat [Membuat pengguna dan grup khusus IAM pertama Anda](#) di Panduan Pengguna IAM.

## Saya ingin mengizinkan orang di luar AWS akun saya untuk mengakses sumber daya Amazon Cognito saya

Anda dapat membuat peran yang dapat digunakan pengguna di akun lain atau orang-orang di luar organisasi Anda untuk mengakses sumber daya Anda. Anda dapat menentukan siapa saja yang dipercaya untuk mengambil peran tersebut. Untuk layanan yang mendukung kebijakan berbasis sumber daya atau daftar kontrol akses (ACL), Anda dapat menggunakan kebijakan tersebut untuk memberi orang akses ke sumber daya Anda.

Untuk mempelajari selengkapnya, periksa referensi berikut:

- Untuk mempelajari apakah Amazon Cognito mendukung fitur-fitur ini, lihat [Cara kerja Amazon Cognito dengan IAM](#)
- Untuk mempelajari cara menyediakan akses ke sumber daya Anda di seluruh sumber daya Akun AWS yang Anda miliki, lihat [Menyediakan akses ke pengguna IAM di pengguna lain Akun AWS yang Anda miliki](#) di Panduan Pengguna IAM.
- Untuk mempelajari cara menyediakan akses ke sumber daya Anda kepada pihak ketiga Akun AWS, lihat [Menyediakan akses yang Akun AWS dimiliki oleh pihak ketiga](#) dalam Panduan Pengguna IAM.
- Untuk mempelajari cara memberikan akses melalui federasi identitas, lihat [Menyediakan akses ke pengguna terautentikasi eksternal \(federasi identitas\)](#) dalam Panduan Pengguna IAM.
- Untuk mempelajari perbedaan antara penggunaan kebijakan peran dan kebijakan berbasis sumber daya untuk akses lintas akun, lihat [Bagaimana peran IAM berbeda dari kebijakan berbasis sumber daya](#) dalam Panduan Pengguna IAM.

## Menggunakan peran terkait layanan untuk Amazon Cognito

[Amazon Cognito menggunakan peran terkait AWS Identity and Access Management layanan \(IAM\)](#). Peran terkait layanan adalah jenis peran IAM yang unik dengan kebijakan kepercayaan yang memungkinkan seseorang Layanan AWS untuk mengambil peran tersebut. Peran terkait layanan telah ditentukan sebelumnya oleh Amazon Cognito dan menyertakan semua izin yang diperlukan layanan untuk memanggil layanan lain atas nama Anda. AWS

Peran tertaut layanan mempermudah pengaturan Amazon Cognito karena Anda tidak perlu menambahkan izin yang diperlukan secara manual. Amazon Cognito menentukan izin peran terkait layanan, dan kecuali ditentukan lain, hanya Amazon Cognito yang dapat menjalankan perannya. Izin

yang ditentukan mencakup kebijakan kepercayaan dan kebijakan izin, serta bahwa kebijakan izin tidak dapat dilampirkan ke entitas IAM lainnya.

Anda dapat menghapus peran yang terhubung dengan layanan hanya setelah menghapus sumber daya terkait terlebih dahulu. Ini melindungi sumber daya Amazon Cognito karena Anda tidak dapat secara tidak sengaja menghapus izin untuk mengakses sumber daya.

Untuk informasi tentang layanan lain yang mendukung peran terkait layanan, lihat [Layanan AWS yang bisa digunakan dengan IAM](#) dan carilah layanan yang memiliki opsi Ya di kolom Peran Terkait Layanan. Pilih Ya bersama tautan untuk melihat dokumentasi peran tertaut layanan untuk layanan tersebut.

## Izin peran terkait layanan untuk Amazon Cognito

Amazon Cognito menggunakan peran terkait layanan berikut:

- `AWSServiceRoleForAmazonCognitoIdpEmailService`— Memungkinkan layanan kumpulan pengguna Amazon Cognito menggunakan identitas Amazon SES Anda untuk mengirim email.
- `AWSServiceRoleForAmazonCognitoIdp`— Memungkinkan kumpulan pengguna Amazon Cognito mempublikasikan peristiwa dan mengonfigurasi titik akhir untuk proyek Amazon Pinpoint Anda.

### `AWSServiceRoleForAmazonCognitoIdpEmailService`

Peran terkait layanan `AWSServiceRoleForAmazonCognitoIdpEmailService` memercayakan layanan berikut untuk menjalankan peran tersebut:

- `email.cognito-idp.amazonaws.com`

Kebijakan izin peran memungkinkan Amazon Cognito menyelesaikan tindakan berikut pada sumber daya yang ditentukan:

Tindakan yang Diizinkan untuk `AWSServiceRoleForAmazonCognitoIdpEmailService`:

- Tindakan: `ses:SendEmail` dan `ses:SendRawEmail`
- Sumber Daya: \*

Kebijakan tersebut menolak kemampuan Amazon Cognito untuk menyelesaikan tindakan berikut pada sumber daya yang ditentukan:

## Tindakan Ditolak

- Tindakan: `ses:List*`
- Sumber Daya: \*

Dengan izin ini, Amazon Cognito dapat menggunakan alamat email terverifikasi Anda di Amazon SES hanya untuk mengirim email kepada pengguna Anda. Amazon Cognito mengirimkan email kepada pengguna Anda saat mereka melakukan tindakan tertentu di aplikasi klien untuk kolam pengguna, seperti mendaftar atau menyetel ulang kata sandi.

Anda harus mengonfigurasi izin untuk mengizinkan entitas IAM (seperti pengguna, grup, atau peran) untuk membuat, menyunting, atau menghapus peran terhubung dengan layanan. Untuk informasi selengkapnya, lihat [Izin peran tertaut layanan](#) dalam Panduan Pengguna IAM.

### AWSServiceRoleForAmazonCognitoIdp

Peran `AWSServiceRoleForAmazonCognitoIdp` terkait layanan mempercayai layanan berikut untuk mengambil peran:

- `email.cognito-idp.amazonaws.com`

Kebijakan izin peran memungkinkan Amazon Cognito menyelesaikan tindakan berikut pada sumber daya yang ditentukan:

### Tindakan yang Diizinkan untuk `AWSServiceRoleForAmazonCognitoIdp`

- Tindakan: `cognito-idp:Describe`
- Sumber Daya: \*

Dengan izin ini, Amazon Cognito dapat memanggil Operasi API Amazon Cognito `Describe` untuk Anda.

#### Note

Bila Anda mengintegrasikan Amazon Cognito dengan Amazon Pinpoint menggunakan `createUserPoolClient` dan `updateUserPoolClient`, izin sumber daya akan ditambahkan ke SLR sebagai kebijakan inline. Kebijakan inline akan memberikan izin `mobiletargeting:UpdateEndpoint` dan `mobiletargeting:PutEvents`. Izin ini

memungkinkan Amazon Cognito mempublikasikan acara dan mengonfigurasi titik akhir untuk proyek Pinpoint yang Anda integrasikan dengan Cognito.

## Membuat peran terkait layanan untuk Amazon Cognito

Anda tidak perlu membuat peran terkait layanan secara manual. Saat mengonfigurasi kumpulan pengguna untuk menggunakan konfigurasi Amazon SES untuk menangani pengiriman email di, API AWS Management Console, atau Amazon Cognito AWS CLI, Amazon Cognito akan membuat peran terkait layanan untuk Anda.

Jika Anda menghapus peran tertaut layanan ini, dan ingin membuatnya lagi, Anda dapat mengulangi proses yang sama untuk membuat kembali peran tersebut di akun Anda. Saat Anda mengonfigurasi kolom pengguna untuk menggunakan konfigurasi Amazon SES Anda untuk menangani pengiriman email, Amazon Cognito menciptakan peran terkait layanan untuk Anda kembali.

Sebelum Amazon Cognito dapat membuat peran ini, izin IAM yang Anda gunakan untuk mengatur kolom pengguna harus menyertakan tindakan `iam:CreateServiceLinkedRole`. Untuk informasi selengkapnya tentang memperbarui izin di IAM, lihat [Mengubah Izin untuk Pengguna IAM](#) di Panduan Pengguna IAM.

## Mengedit peran terkait layanan untuk Amazon Cognito

Anda tidak dapat mengedit `AmazonCognitoIdp` atau peran `AmazonCognitoIdpEmailService` terkait layanan di AWS Identity and Access Management. Setelah membuat peran terkait layanan, Anda tidak dapat mengubah nama peran karena berbagai entitas mungkin mereferensikan peran tersebut. Namun, Anda dapat menyunting penjelasan peran menggunakan IAM. Untuk informasi selengkapnya, lihat [Mengedit peran tertaut layanan](#) dalam Panduan Pengguna IAM.


## Menghapus peran terkait layanan untuk Amazon Cognito

Jika Anda tidak perlu lagi menggunakan fitur atau layanan yang memerlukan peran terkait layanan, kami merekomendasikan Anda menghapus peran tersebut. Jika Anda menghapus peran, Anda hanya mempertahankan entitas yang Amazon Cognito secara aktif memonitor atau memelihara. Sebelum dapat menghapus `AmazonCognitoIdp` atau peran `AmazonCognitoIdpEmailService` terkait layanan, Anda harus melakukan salah satu hal berikut untuk setiap kumpulan pengguna yang menggunakan peran tersebut:

- Hapus kolom pengguna.

- Perbarui pengaturan email di kolom pengguna untuk menggunakan fungsionalitas email default. Pengaturan default tidak menggunakan peran terkait layanan.

Ingatlah untuk melakukan tindakan di masing-masing Wilayah AWS dengan kumpulan pengguna yang menggunakan peran tersebut.

 Note

Jika layanan Amazon Cognito menggunakan peran saat Anda mencoba untuk menghapus sumber daya, maka penghapusan tersebut kemungkinan gagal. Jika hal itu terjadi, tunggu beberapa menit dan coba lagi.

Untuk menghapus kolam pengguna Amazon Cognito

1. Masuk ke AWS Management Console dan buka konsol Amazon Cognito di. <https://console.aws.amazon.com/cognito>
2. Pilih Kelola Kolam Pengguna.
3. Pada halaman Kolam Pengguna Anda, pilih kolam pengguna yang ingin Anda hapus.
4. Pilih Hapus kolam.
5. Di jendela Hapus kolam pengguna, ketik **delete**, dan pilih Hapus kolam.

Untuk memperbarui kumpulan pengguna Amazon Cognito untuk menggunakan fungsionalitas email default

1. Masuk ke AWS Management Console dan buka konsol Amazon Cognito di. <https://console.aws.amazon.com/cognito>
2. Pilih Kelola Kolam Pengguna.
3. Pada halaman Kolam Pengguna Anda, pilih kolam pengguna yang ingin Anda perbarui.
4. Di menu navigasi di sebelah kiri, pilih Kustomisasi pesan.
5. Di bawah Apakah Anda ingin mengirim email melalui Konfigurasi Amazon SES Anda?, pilih Tidak - Gunakan Cognito (Default).
6. Setelah selesai mengatur opsi akun email Anda, pilih Simpan perubahan.

Untuk menghapus peran terkait layanan secara manual menggunakan IAM

Gunakan konsol IAM, the AWS CLI, atau AWS API untuk menghapus AmazonCognitoIdp atau peran yang ditautkan AmazonCognitoIdpEmailService layanan. Untuk informasi selengkapnya, lihat [Menghapus peran tertaut layanan](#) dalam Panduan Pengguna IAM.

## Wilayah yang Didukung untuk peran terkait layanan Amazon Cognito

Amazon Cognito mendukung peran terkait layanan di semua Wilayah AWS tempat layanan tersedia. Untuk informasi lebih lanjut, lihat [Wilayah AWS dan Titik Akhir](#).

## Pencatatan dan pemantauan di Amazon Cognito

Pemantauan adalah bagian penting dalam menjaga keandalan, ketersediaan, dan kinerja Amazon Cognito dan solusi Anda yang lain AWS . Amazon Cognito saat ini mendukung hal berikut Layanan AWS sehingga Anda dapat memantau organisasi Anda dan aktivitas yang terjadi di dalamnya.

- **AWS CloudTrail** — Dengan CloudTrail Anda dapat menangkap panggilan API dari konsol Amazon Cognito dan dari panggilan kode ke operasi Amazon Cognito API. Misalnya, ketika pengguna mengautentikasi, CloudTrail dapat merekam detail seperti alamat IP dalam permintaan, siapa yang membuat permintaan, dan kapan itu dibuat.
- **CloudWatch Log Amazon** — Dengan CloudWatch Log, Anda dapat mengirim log aktivitas pengguna yang berbutir halus ke grup log. Misalnya, Anda dapat meninjau log aktivitas pengguna terperinci untuk memecahkan masalah pengiriman pesan email dan SMS ke pengguna Anda.
- **Amazon CloudWatch Metrics** — Dengan CloudWatch metrik, Anda dapat memantau, melaporkan, dan mengambil tindakan otomatis jika terjadi peristiwa dalam waktu dekat. Misalnya, Anda dapat membuat CloudWatch dasbor pada metrik yang disediakan untuk memantau kumpulan pengguna Amazon Cognito, atau Anda dapat CloudWatch membuat alarm pada metrik yang disediakan untuk memberi tahu Anda tentang pelanggaran ambang batas yang ditetapkan.
- **Amazon CloudWatch Logs Insights** — Dengan Wawasan CloudWatch Log, Anda dapat mengonfigurasi CloudTrail untuk mengirim acara CloudWatch untuk memantau file log Amazon CloudTrail Cognito.

### Topik

- [Biaya pemantauan](#)
- [Melacak kuota dan penggunaan di CloudWatch dan Service Quotas](#)
- [Mencatat panggilan API Amazon Cognito dengan AWS CloudTrail](#)

## Biaya pemantauan

Amazon Cognito mengenakan biaya untuk dimensi penggunaan Anda berikut.

- Kumpulan pengguna pengguna aktif bulanan (MAUs)
- UI kumpulan pengguna yang masuk dengan federasi OIDC atau SAMP
- MAUs di kumpulan pengguna dengan fitur keamanan canggih
- Klien aplikasi kumpulan pengguna aktif dan volume permintaan untuk otorisasi mesin ke mesin (M2M) dengan hibah kredensial klien
- Penggunaan yang dibeli di atas kuota default untuk beberapa kategori API kumpulan pengguna

Selain itu, fitur kumpulan pengguna Anda seperti pesan email, pesan SMS, dan pemicu Lambda dapat menimbulkan biaya dalam layanan dependen. Untuk ikhtisar selengkapnya, lihat [Harga Amazon Cognito](#).

### Melihat dan mengantisipasi biaya

Anda dapat melihat dan melaporkan AWS biaya Anda di [AWS Billing and Cost Management konsol](#). Anda dapat menemukan tagihan terbaru untuk Amazon Cognito di bagian Penagihan dan pembayaran. Di bawah Tagihan, Biaya berdasarkan layanan, filter Cognito untuk melihat penggunaan Anda. Untuk informasi selengkapnya, lihat [Melihat tagihan Anda](#) dalam Panduan Pengguna AWS Billing .

Untuk memantau tingkat permintaan API, tinjau metrik Utilisasi di konsol Service Quotas. Misalnya, permintaan kredensial klien ditampilkan sebagai Tingkat permintaan. ClientAuthentication Dalam tagihan Anda, permintaan ini terkait dengan klien aplikasi yang memproduksinya. [Dengan informasi ini, Anda dapat mengalokasikan biaya secara adil kepada penyewa dalam arsitektur multi-penyewa.](#)

Untuk mendapatkan hitungan permintaan M2M untuk jangka waktu tertentu, Anda juga dapat mengirim [AWS CloudTrail peristiwa ke CloudWatch Log untuk analisis](#). Kueri CloudTrail acara Anda untuk Token\_POST acara dengan hibah kredensial klien. Query CloudWatch Insights berikut mengembalikan hitungan ini.

```
filter eventName = "Token_POST" and @message like '"grant_type":["client_credentials"]'
| stats count(*)
```



## Mengelola biaya

Amazon Cognito menagih berdasarkan jumlah pengguna, penggunaan fitur, dan volume permintaan. Berikut ini adalah beberapa tips untuk mengelola biaya di Amazon Cognito,

Jangan aktifkan pengguna yang tidak aktif

Operasi umum yang membuat pengguna aktif adalah masuk, mendaftar, dan mengatur ulang kata sandi. Untuk daftar yang lebih lengkap, lihat [Pengguna aktif bulanan](#). Amazon Cognito tidak menghitung pengguna yang tidak aktif terhadap tagihan Anda. Hindari operasi apa pun yang mengatur pengguna aktif. Alih-alih operasi [AdminGetUserAPI](#), kueri pengguna dengan [ListUsers](#) operasi. Jangan melakukan pengujian administratif volume tinggi dari operasi kumpulan pengguna dengan pengguna yang tidak aktif.

Tautkan pengguna federasi

[Pengguna yang masuk dengan penyedia identitas SAMP 2.0 atau OpenID Connect \(OIDC\) memiliki biaya lebih tinggi daripada pengguna lokal](#). Anda dapat [menautkan pengguna ini ke profil pengguna lokal](#). Pengguna yang tertaut dapat masuk sebagai pengguna lokal dengan atribut dan akses yang disertakan dengan pengguna federasi mereka. Pengguna dari SAMP atau OIDC IdPs yang, dalam waktu satu bulan, hanya masuk dengan akun lokal yang ditautkan akan ditagih sebagai pengguna lokal.

Kelola tarif permintaan

Jika kumpulan pengguna Anda mendekati batas atas kuota Anda, Anda dapat mempertimbangkan untuk membeli kapasitas tambahan untuk menangani volume. Anda mungkin dapat mengurangi volume permintaan dalam aplikasi Anda. Untuk informasi selengkapnya, lihat [Optimalkan tarif permintaan untuk batas kuota](#).

Minta token baru hanya jika Anda membutuhkannya

Otorisasi mesin ke mesin (M2M) dengan hibah kredensial klien dapat mencapai volume permintaan token yang tinggi. Setiap permintaan token baru berpengaruh pada kuota request-rate Anda dan ukuran tagihan Anda. Untuk mengoptimalkan biaya, sertakan pengaturan kedaluwarsa token dan penanganan token dalam desain aplikasi Anda.

- [Cache token akses](#) sehingga ketika aplikasi Anda meminta token baru, ia menerima versi cache dari token yang dikeluarkan sebelumnya. Saat Anda menerapkan metode ini, proxy caching Anda bertindak sebagai penjaga terhadap aplikasi yang meminta token akses tanpa menyadari

kedaluwarsa token yang diperoleh sebelumnya. Token caching sangat ideal untuk layanan mikro berumur pendek seperti fungsi Lambda dan wadah Docker.

- Terapkan mekanisme penanganan token di aplikasi Anda yang memperhitungkan kedaluwarsa token. Jangan meminta token baru sampai token sebelumnya kedaluwarsa. Evaluasi kebutuhan kerahasiaan dan ketersediaan setiap aplikasi dan konfigurasi klien aplikasi kumpulan pengguna untuk mengeluarkan token akses dengan masa berlaku yang sesuai. Durasi token khusus berfungsi paling baik dengan API dan server yang berumur lebih lama yang dapat mengelola frekuensi permintaan kredensial secara terus-menerus.

Hapus klien aplikasi kredensial klien yang tidak digunakan

Tagihan otorisasi M2M berdasarkan dua faktor: tingkat permintaan token dan jumlah klien aplikasi yang memberikan kredensial klien. Saat klien aplikasi untuk otorisasi M2M tidak digunakan, hapus atau hapus otorisasi mereka untuk mengeluarkan kredensial klien. Untuk informasi selengkapnya tentang mengelola konfigurasi klien aplikasi, lihat [Klien aplikasi kumpulan pengguna](#).

Kelola keamanan tingkat lanjut

Saat Anda mengonfigurasi [fitur keamanan lanjutan](#) di kumpulan pengguna, tingkat penagihan keamanan lanjutan berlaku untuk semua UI di kumpulan pengguna. Jika Anda memiliki pengguna yang tidak memerlukan fitur keamanan tingkat lanjut, pisahkan mereka ke kumpulan pengguna lain.

## Melacak kuota dan penggunaan di CloudWatch dan Service Quotas

Anda dapat memantau kumpulan pengguna Amazon Cognito menggunakan Amazon CloudWatch atau menggunakan Service Quotas. Anda juga dapat memantau penggunaan kumpulan identitas di Service Quotas. CloudWatch mengumpulkan data mentah dan memprosesnya menjadi metrik yang dapat dibaca, mendekati waktu nyata. Di CloudWatch, Anda dapat menyetel alarm yang mengawasi ambang batas tertentu dan mengirim pemberitahuan atau mengambil tindakan saat ambang batas tersebut terpenuhi. Untuk membuat CloudWatch alarm untuk kuota layanan, lihat [Membuat CloudWatch alarm](#). Metrik Amazon Cognito tersedia pada interval lima menit. Untuk informasi selengkapnya tentang periode retensi di CloudWatch, kunjungi [halaman CloudWatch FAQ Amazon](#).

Anda dapat menggunakan Service Quotas untuk melihat dan mengelola kumpulan pengguna Amazon Cognito dan penggunaan kuota kumpulan identitas. Konsol Service Quotas memiliki tiga fitur: melihat kuota layanan, meminta peningkatan kuota layanan, dan melihat pemanfaatan saat ini. Anda dapat menggunakan fitur pertama untuk melihat kuota dan melihat apakah kuota dapat

disesuaikan. Anda dapat menggunakan fitur kedua untuk meminta peningkatan Service Quotas. Anda dapat menggunakan fitur terakhir untuk melihat utilisasi kuota. Fitur ini hanya tersedia setelah akun Anda aktif untuk sementara waktu. Untuk informasi selengkapnya tentang melihat kuota di konsol Service Quotas, lihat [Service Quotas](#).

#### Note

Metrik Amazon Cognito tersedia pada interval 5 menit. Untuk informasi selengkapnya tentang periode retensi di CloudWatch, kunjungi [halaman CloudWatch FAQ Amazon](#).

Jika Anda masuk ke akun Akun AWS yang diatur sebagai akun pemantauan dalam pengamatan CloudWatch lintas akun, Anda dapat menggunakan akun pemantauan tersebut untuk memvisualisasikan kuota layanan dan menyetel alarm untuk metrik di akun sumber yang ditautkan ke akun pemantauan tersebut. Untuk informasi lebih lanjut, lihat [CloudWatch observabilitas lintas akun](#).

#### Topik

- [Mencatat aktivitas tambahan dari kumpulan pengguna Amazon Cognito](#)
- [Metrik untuk kolam pengguna Amazon Cognito](#)
- [Dimensi untuk kolam pengguna Amazon Cognito](#)
- [Gunakan konsol Service Quotas untuk melacak metrik](#)
- [Gunakan CloudWatch konsol untuk melacak metrik](#)
- [Buat CloudWatch alarm untuk kuota](#)

## Mencatat aktivitas tambahan dari kumpulan pengguna Amazon Cognito

Anda dapat mengonfigurasi kumpulan pengguna untuk mengirim log terperinci dari beberapa aktivitas tambahan ke grup CloudWatch log. Log ini memiliki perincian yang lebih halus daripada yang ada di dalamnya AWS CloudTrail, dan dapat berguna untuk memecahkan masalah kumpulan pengguna Anda. Saat Anda mengaktifkan fitur ini, Anda dapat memilih grup log tempat Anda ingin Amazon Cognito mengirim log. Pencatatan aktivitas pengguna berguna saat Anda ingin mengetahui status pesan email dan SMS yang dikirimkan oleh kumpulan pengguna Anda dengan Amazon SNS dan Amazon SES.

Saat ini, Anda hanya dapat mengirimkan log pemberitahuan pengguna tingkat Kesalahan dari kumpulan pengguna Anda.

Pencatatan terperinci tidak menggantikan atau mengubah fungsi log berikut dari kumpulan pengguna.

1. CloudTrail log aktivitas pengguna rutin seperti sign-up dan sign-in.
2. Analisis aktivitas pengguna dalam skala dengan CloudWatch metrik.

Secara terpisah, Anda juga dapat menemukan log dari [pekerjaan impor pengguna](#) dan [pemicu Lambda](#) di Log. CloudWatch Amazon Cognito dan Lambda menyimpan log ini di grup log yang berbeda dari yang Anda tentukan untuk log aktivitas terperinci.

Anda dapat mengonfigurasi log aktivitas terperinci dengan API kumpulan pengguna Amazon Cognito dalam permintaan [SetLogDeliveryConfiguration](#) API. Anda dapat melihat konfigurasi logging kumpulan pengguna dalam permintaan [GetLogDeliveryConfiguration](#) API.

Anda harus mengotorisasi permintaan ini dengan AWS kredensi yang memiliki izin berikut.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ManageUserPoolLogs",
      "Action": [
        "cognito-idp:SetLogDeliveryConfiguration",
        "cognito-idp:GetLogDeliveryConfiguration",
      ],
      "Resource": [
        "*"
      ],
      "Effect": "Allow"
    },
    {
      "Sid": "CognitoLog",
      "Action": [
        "logs:CreateLogDelivery",
        "logs:GetLogDelivery",
        "logs:UpdateLogDelivery",
        "logs>DeleteLogDelivery",
        "logs:ListLogDeliveries"
      ],
      "Resource": [
        "*"
      ],
    },
  ],
}
```

```

    "Effect": "Allow"
  },
  {
    "Sid": "CognitoLoggingCWL",
    "Action": [
      "logs:PutResourcePolicy",
      "logs:DescribeResourcePolicies",
      "logs:DescribeLogGroups"
    ],
    "Resource": [
      "*"
    ],
    "Effect": "Allow"
  }
]
}

```

Berikut ini adalah contoh peristiwa dari kumpulan pengguna. Skema log ini dapat berubah sewaktu-waktu. Beberapa bidang mungkin dicatat dengan nilai null.

```

{
  "eventTimestamp": "1687297330677",
  "eventSource": "USER_NOTIFICATION",
  "logLevel": "ERROR",
  "message": {
    "details": "String"
  },
  "logSourceId": {
    "userPoolId": "String"
  }
}

```

Pengiriman log dari Amazon Cognito adalah upaya terbaik. Volume log yang diberikan oleh kumpulan pengguna Anda, dan kuota layanan Anda untuk CloudWatch Log, dapat memengaruhi pengiriman log.

CloudWatch Biaya log berlaku saat pengiriman log diaktifkan. Untuk informasi selengkapnya, lihat [Log Terjual](#) di CloudWatch Harga Amazon.

Untuk mengirim log ke grup log dengan kebijakan sumber daya berukuran lebih besar dari 5120 karakter, konfigurasi grup log dengan jalur yang dimulai dengan `/aws/vendedlogs`. Untuk informasi selengkapnya, lihat [Mengaktifkan logging dari AWS layanan tertentu](#).

## Metrik untuk kolam pengguna Amazon Cognito

Tabel berikut mencantumkan metrik yang tersedia untuk kolam pengguna Amazon Cognito. Namespace Amazon CloudWatch metrik untuk Amazon Cognito adalah `AWS/Cognito`. Untuk informasi selengkapnya, lihat [Ruang nama](#) di CloudWatch Panduan Pengguna Amazon.

### Note

Metrik yang tidak memiliki titik data baru dalam dua minggu terakhir tidak muncul di konsol. Nama metrik atau nama dimensi juga tidak muncul saat Anda memasukkan namanya di kotak pencarian di tab Semua metrik di konsol. Selain itu, metrik tidak dikembalikan dalam hasil perintah daftar-metrik. Cara terbaik untuk mengambil metrik ini adalah dengan `get-metric-statistics` perintah `get-metric-data` or di CLI AWS .

Metrik	Deskripsi
<code>SignUpSuccesses</code>	<p>Menyediakan jumlah total permintaan pendaftaran pengguna yang berhasil dibuat ke kolam pengguna Amazon Cognito. Permintaan pendaftaran pengguna yang berhasil menghasilkan nilai 1, sedangkan permintaan yang gagal menghasilkan nilai 0. Permintaan throttling juga dianggap sebagai permintaan gagal, dan karenanya permintaan throttling juga akan menghasilkan hitungan 0.</p> <p>Untuk menemukan persentase permintaan pendaftaran pengguna yang berhasil, gunakan statistik <code>Average</code> pada metrik ini. Untuk menghitung jumlah total permintaan pendaftaran pengguna, gunakan statistik <code>Sample Count</code> pada metrik ini. Untuk menghitung jumlah total permintaan pendaftaran pengguna berhasil, gunakan statistik <code>Sum</code> pada metrik ini. Untuk menghitung jumlah total permintaan pendaftaran pengguna yang gagal, gunakan</p>

Metrik	Deskripsi
	<p>CloudWatch Math ekspresi dan kurangi Sum statistik dari Sample Count statistik.</p> <p>Metrik ini diterbitkan untuk setiap kolom pengguna untuk setiap klien kolom pengguna. Dalam kasus ketika pendaftaran pengguna dilakukan oleh admin, metrik diterbitkan dengan klien kolom pengguna sebagai Admin.</p> <p>Perhatikan bahwa metrik ini tidak dipancarkan untuk kasus <a href="#">impor Pengguna</a> dan <a href="#">migrasi Pengguna</a>.</p> <p>Dimensi metrik: UserPool, UserPoolClient</p> <p>Unit: Jumlah</p>

Metrik	Deskripsi
SignUpThrottles	<p>Menyediakan jumlah total permintaan pendaftaran pengguna throttling yang dibuat ke kolam pengguna Amazon Cognito. Hitungan 1 diterbitkan setiap kali permintaan pendaftaran pengguna throttling.</p> <p>Untuk menghitung jumlah total permintaan pendaftaran pengguna throttling, gunakan statistik Sum pada metrik ini.</p> <p>Metrik ini diterbitkan untuk setiap kolam pengguna untuk setiap klien. Dalam kasus ketika permintaan throttling dibuat oleh administrator, metrik diterbitkan dengan klien kolam pengguna sebagai Admin.</p> <p>Dimensi metrik: <code>UserPool</code>, <code>UserPoolClient</code></p> <p>Unit: Jumlah</p>



Metrik	Deskripsi
SignInSuccesses	<p>Menyediakan jumlah total permintaan autentikasi pengguna yang berhasil dibuat ke kolom pengguna Amazon Cognito. Autentikasi pengguna dianggap berhasil ketika token autentikasi dikeluarkan untuk pengguna. Sebuah autentikasi berhasil menghasilkan nilai 1, sedangkan permintaan gagal menghasilkan nilai 0. Permintaan throttling juga dianggap sebagai permintaan gagal, dan karenanya permintaan throttling juga akan menghasilkan hitungan 0.</p> <p>Untuk menemukan persentase permintaan autentikasi pengguna yang berhasil, gunakan statistik <code>Average</code> pada metrik ini. Untuk menghitung jumlah permintaan autentikasi pengguna, gunakan statistik <code>Sample Count</code> pada metrik ini. Untuk menghitung jumlah total permintaan autentikasi pengguna berhasil, gunakan statistik <code>Sum</code> pada metrik ini. Untuk menghitung jumlah total permintaan autentikasi pengguna yang gagal, gunakan <code>CloudWatch Math</code> ekspresi dan kurangi <code>Sum</code> statistik dari statistik <code>Sample Count</code>.</p> <p>Metrik ini diterbitkan untuk setiap kolom pengguna untuk setiap klien. Jika klien kolom pengguna yang tidak valid diberikan permintaan, nilai klien kolom pengguna yang sesuai dalam metrik berisi nilai tetap <code>Invalid</code>, bukan nilai tidak valid sebenarnya yang dikirim dalam permintaan.</p> <p>Perhatikan bahwa permintaan untuk me-refresh token Amazon Cognito tidak disertakan dalam</p>

Metrik	Deskripsi
	<p>metrik ini. Ada metrik terpisah untuk menyediakan statistik token Refresh.</p> <p>Dimensi metrik: <code>UserPool</code>, <code>UserPoolClient</code></p> <p>Unit: Jumlah</p>
<code>SignInThrottles</code>	<p>Menyediakan jumlah total permintaan autentikasi pengguna throttling yang dibuat ke kolom pengguna Amazon Cognito. Hitungan 1 diterbitkan setiap kali permintaan autentikasi di-throttling.</p> <p>Untuk menghitung jumlah total permintaan autentikasi pengguna yang di-throttling, gunakan statistikSum untuk metrik ini.</p> <p>Metrik ini diterbitkan untuk setiap kolom pengguna untuk setiap klien. Jika klien kolom pengguna yang tidak valid diberikan permintaan, nilai klien kolom pengguna yang sesuai dalam metrik berisi nilai tetap <code>Invalid</code>, bukan nilai tidak valid sebenarnya yang dikirim dalam permintaan.</p> <p>Permintaan untuk me-refresh token Amazon Cognito tidak disertakan dalam metrik ini. Ada metrik terpisah untuk menyediakan statistik token Refresh.</p> <p>Dimensi metrik: <code>UserPool</code>, <code>UserPoolClient</code></p> <p>Unit: Jumlah</p>

Metrik	Deskripsi
TokenRefreshSuccesses	<p>Menyediakan jumlah total permintaan yang berhasil untuk me-refresh token Amazon Cognito yang dibuat ke kolom pengguna Amazon Cognito. Permintaan token Amazon Cognito refresh yang berhasil menghasilkan nilai 1, sedangkan permintaan yang gagal menghasilkan nilai 0. Permintaan throttling juga dianggap sebagai permintaan gagal, dan karenanya permintaan throttling juga akan menghasilkan hitungan 0.</p> <p>Untuk menemukan persentase permintaan yang berhasil untuk me-refresh token Amazon Cognito, gunakan statistik <code>Average</code> pada metrik ini. Untuk menghitung jumlah total permintaan untuk me-refresh token Amazon Cognito, gunakan statistik <code>Sample Count</code> pada metrik ini. Untuk menghitung jumlah total permintaan yang berhasil untuk me-refresh token Amazon Cognito, gunakan statistik <code>Sum</code> pada metrik ini. Untuk menghitung jumlah total permintaan yang gagal untuk menyegarkan token Amazon Cognito, gunakan <code>CloudWatch Math</code> ekspresi dan kurangi <code>Sum</code> statistik dari statistik <code>Sample Count</code></p> <p>Metrik ini dipublikasikan per setiap klien kolom pengguna. Jika klien kolom pengguna tidak valid dalam permintaan, nilai klien kolom pengguna berisi nilai tetap <code>Invalid</code>.</p> <p>Dimensi metrik: <code>UserPool</code>, <code>UserPoolClient</code></p> <p>Unit: Jumlah</p>

Metrik	Deskripsi
TokenRefreshThrottles	<p>Menyediakan jumlah permintaan yang dibatasi untuk me-refresh token Amazon Cognito yang dibuat ke kolom pengguna Amazon Cognito. Hitungan 1 diterbitkan setiap kali permintaan token refresh Amazon Cognito dibatasi.</p> <p>Untuk menghitung jumlah total permintaan yang dibatasi untuk menyegarkan token Amazon Cognito, gunakan statistik Sum untuk metrik ini.</p> <p>Metrik ini diterbitkan untuk setiap kolom pengguna untuk setiap klien. Jika klien kolom pengguna yang tidak valid diberikan permintaan, nilai klien kolom pengguna yang sesuai dalam metrik berisi nilai tetap <code>Invalid</code>, bukan nilai tidak valid sebenarnya yang dikirim dalam permintaan.</p> <p>Dimensi metrik: <code>UserPool</code>, <code>UserPoolClient</code></p> <p>Unit: Jumlah</p>

Metrik	Deskripsi
FederationSuccesses	<p>Memberikan jumlah total permintaan federasi identitas yang berhasil ke kumpulan pengguna Amazon Cognito. Federasi identitas dianggap berhasil ketika Amazon Cognito mengeluarkan token otentikasi kepada pengguna. Sebuah permintaan federasi identitas berhasil menghasilkan nilai 1, sedangkan permintaan gagal menghasilkan nilai 0. Permintaan dan permintaan yang dibatasi yang menghasilkan kode otorisasi tetapi tidak ada token yang menghasilkan nilai 0.</p> <p>Untuk menemukan persentase permintaan federasi identitas yang berhasil, gunakan statistik <code>Average</code> pada metrik ini. Untuk menghitung jumlah total permintaan federasi identitas, gunakan statistik <code>Sample Count</code> pada metrik ini. Untuk menghitung jumlah total permintaan federasi identitas yang berhasil, gunakan statistik <code>Sum</code> pada metrik ini. Untuk menghitung jumlah total permintaan federasi identitas yang gagal, gunakan <code>CloudWatch Math</code> ekspresi dan kurangi <code>Sum</code> statistik dari <code>Sample Count</code> statistik.</p> <p>Dimensi metrik: <code>UserPool</code>, <code>UserPoolClient</code>, <code>IdentityProvider</code></p> <p>Unit: Jumlah</p>

Metrik	Deskripsi
FederationThrottles	<p>Menyediakan jumlah total permintaan federasi identitas yang dibatasi ke kolam pengguna Amazon Cognito. Hitungan 1 diterbitkan setiap kali permintaan federasi identitas dibatasi.</p> <p>Untuk menghitung jumlah total permintaan federasi identitas yang dibatasi, gunakan statistik Sum untuk metrik ini.</p> <p>Dimensi metrik: <code>UserPool</code>, <code>UserPoolClient</code>, <code>IdentityProvider</code></p> <p>Unit: Jumlah</p>
CallCount	<p>Menyediakan jumlah panggilan pelanggan yang dibuat terkait dengan kategori. Metrik ini mencakup semua panggilan, seperti panggilan terbatas, panggilan gagal, dan panggilan sukses.</p> <p>Metrik ini tersedia di Penggunaan namespace <code>.</code></p> <p>Kuota kategori diberlakukan untuk setiap AWS akun di semua kumpulan pengguna di akun dan Wilayah.</p> <p>Anda dapat menghitung jumlah total panggilan dalam kategori menggunakan statistik Sum untuk metrik ini.</p> <p>Dimensi metrik: <code>Layanan</code>, <code>Jenis</code>, <code>Sumber Daya</code>, <code>Kelas</code></p> <p>Unit: Jumlah</p>

Metrik	Deskripsi
ThrottleCount	<p>Menyediakan jumlah panggilan yang dibatasi yang terkait dengan kategori.</p> <p>Metrik ini tersedia di Penggunaan nameSpace .</p> <p>Metrik ini dipublikasikan di tingkat akun.</p> <p>Anda dapat menghitung jumlah total panggilan dalam kategori, menggunakan statistik Sum untuk metrik ini.</p> <p>Dimensi metrik: Layanan, Jenis, Sumber Daya, Kelas</p> <p>Unit: Jumlah</p>

## Dimensi untuk kolam pengguna Amazon Cognito

Dimensi berikut digunakan untuk menyempurnakan metrik penggunaan yang diterbitkan oleh Amazon Cognito. Dimensi hanya berlaku untuk metrik CallCount dan ThrottleCount .

Dimensi	Deskripsi
Layanan	Nama AWS layanan yang berisi sumber daya. Untuk metrik penggunaan Amazon Cognito, nilai untuk dimensi ini adalah Cognito user pool.
Jenis	Jenis entitas yang dilaporkan. Satu-satunya nilai yang valid untuk metrik penggunaan Amazon Cognito adalah API.
Sumber Daya	Jenis sumber daya yang sedang berjalan. Satu-satunya nilai yang valid adalah nama kategori.

Dimensi	Deskripsi
Kelas	Kelas sumber daya yang ditelusuri. Amazon Cognito tidak menggunakan dimensi kelas.

## Gunakan konsol Service Quotas untuk melacak metrik

Anda dapat melihat dan mengelola kumpulan pengguna Amazon Cognito dan kuota kumpulan identitas dari lokasi pusat dengan Service Quotas. Anda dapat menggunakan konsol Service Quotas untuk melihat detail tentang kuota tertentu, memantau penggunaan kuota, dan meminta kenaikan kuota. Untuk beberapa jenis kuota, Anda dapat membuat CloudWatch alarm untuk melacak penggunaan kuota Anda. Untuk mempelajari lebih lanjut tentang metrik Amazon Cognito yang dapat Anda lacak, lihat. [Lacak penggunaan kuota](#)

Untuk melihat kumpulan pengguna Amazon Cognito dan pemanfaatan kuota layanan kumpulan identitas, selesaikan langkah-langkah berikut.

1. Buka [Konsol Service Quotas](#).
2. Di panel navigasi, pilih Layanan AWS .
3. Dari daftar AWS layanan, cari dan pilih kumpulan pengguna Amazon Cognito atau Identitas Federasi Amazon Cognito. Halaman service quotas akan muncul.
4. Pilih kuota yang mendukung CloudWatch pemantauan. Misalnya, pilih `Rate of UserAuthentication requests` di kumpulan pengguna Amazon Cognito.
5. Gulir ke bawah ke Pemantauan. Bagian ini hanya muncul untuk kuota yang mendukung CloudWatch pemantauan.
6. Di Pemantauan Anda dapat melihat utilisasi kuota layanan saat ini dalam grafik.
7. Di Pemantauan pilih satu jam, tiga jam, dua belas jam, satu hari, tiga hari, atau satu minggu.
8. Pilih area manapun di dalam grafik untuk melihat persentase penggunaan kuota layanan. Dari sini, Anda dapat menambahkan grafik ke dasbor Anda atau menggunakan menu tindakan untuk memilih Lihat dalam metrik, yang akan membawa Anda ke metrik terkait di CloudWatch konsol.

## Gunakan CloudWatch konsol untuk melacak metrik

Anda dapat melacak dan mengumpulkan metrik kumpulan pengguna Amazon Cognito menggunakan CloudWatch CloudWatch Dasbor akan menampilkan metrik tentang setiap AWS



layanan yang Anda gunakan. Anda dapat menggunakan CloudWatch untuk membuat alarm metrik. Alarm dapat diatur untuk mengirim Anda pemberitahuan atau membuat perubahan ke sumber daya tertentu yang Anda pantau. Untuk melihat metrik kuota layanan CloudWatch, selesaikan langkah-langkah berikut.

1. Buka [konsol CloudWatch](#).
2. Pada panel navigasi, silakan pilih Metrik.
3. Di Semua metrik pilih metrik dan dimensi.
4. Pilih kotak centang di samping metrik. Metrik akan muncul dalam grafik.

#### Note

Metrik yang tidak memiliki titik data baru dalam dua minggu terakhir tidak muncul di konsol. Metrik juga tidak muncul ketika Anda memasukkan nama metrik atau nama dimensi mereka di kotak pencarian dalam tab Semua metrik di konsol, dan tidak dikembalikan di hasil perintah metrik daftar. Cara terbaik untuk mengambil metrik ini adalah dengan perintah `get-metric-data` atau `get-metric-statistics` di CLI AWS .

## Buat CloudWatch alarm untuk kuota

Amazon Cognito menyediakan metrik CloudWatch penggunaan yang sesuai dengan kuota AWS layanan untuk dan API. `CallCount` `ThrottleCount` Untuk informasi selengkapnya tentang melacak penggunaan di CloudWatch, lihat [Lacak penggunaan kuota](#).

Di konsol Service Quotas, Anda dapat membuat alarm yang memperingatkan Anda saat penggunaan Anda mendekati kuota layanan. Untuk mempelajari cara mengatur CloudWatch alarm menggunakan konsol Service Quotas, lihat Service [Quotas](#) dan alarm. CloudWatch

## Mencatat panggilan API Amazon Cognito dengan AWS CloudTrail

Amazon Cognito terintegrasi dengan AWS CloudTrail, layanan yang menyediakan catatan tindakan yang diambil oleh pengguna, peran, atau AWS layanan di Amazon Cognito. CloudTrail menangkap subset panggilan API untuk Amazon Cognito sebagai peristiwa, termasuk panggilan dari konsol Amazon Cognito dan dari panggilan kode ke operasi Amazon Cognito API. Jika Anda membuat jejak, Anda dapat memilih untuk mengirimkan CloudTrail acara ke bucket Amazon S3, termasuk acara untuk Amazon Cognito. Jika Anda tidak mengonfigurasi jejak, Anda masih dapat melihat

peristiwa terbaru di CloudTrail konsol dalam Riwayat acara. Dengan menggunakan informasi yang dikumpulkan oleh CloudTrail, Anda dapat menentukan permintaan yang dibuat untuk Amazon Cognito, alamat IP dari mana permintaan dibuat, siapa yang membuat permintaan, kapan dibuat, dan detail tambahan.

Untuk mempelajari selengkapnya CloudTrail, termasuk cara mengonfigurasi dan mengaktifkannya, lihat [Panduan AWS CloudTrail Pengguna](#).

Anda juga dapat membuat CloudWatch alarm Amazon untuk CloudTrail acara tertentu. Misalnya, Anda dapat mengatur CloudWatch untuk memicu alarm jika konfigurasi kumpulan identitas diubah. Untuk informasi selengkapnya, lihat [Membuat CloudWatch alarm untuk CloudTrail acara: Contoh](#).

## Topik

- [Informasi Amazon Cognito di CloudTrail](#)
- [Memahami peristiwa masuk Amazon Cognito](#)
- [Menganalisis CloudTrail peristiwa Amazon Cognito dengan Amazon CloudWatch Logs Insights](#)

## Informasi Amazon Cognito di CloudTrail

CloudTrail dihidupkan saat Anda membuat Akun AWS. Ketika aktivitas peristiwa yang didukung terjadi di Amazon Cognito, aktivitas tersebut direkam dalam suatu CloudTrail peristiwa bersama dengan peristiwa AWS layanan lainnya dalam riwayat Acara. Anda dapat melihat, mencari, dan mengunduh acara terbaru di AWS akun Anda. Untuk informasi selengkapnya, lihat [Melihat peristiwa dengan riwayat CloudTrail acara](#).

Untuk catatan peristiwa yang sedang berlangsung di AWS akun Anda, termasuk acara untuk Amazon Cognito, buat jejak. CloudTrail Jejak mengirimkan file log ke bucket Amazon S3. Secara default, ketika Anda membuat jejak di konsol, jejak diterapkan ke semua Region. 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 untuk membuat jejak](#)
- [CloudTrail layanan dan integrasi yang didukung](#)
- [Mengkonfigurasi notifikasi SNS amazon untuk CloudTrail](#)

- [Menerima file CloudTrail log dari beberapa wilayah](#) dan [Menerima file CloudTrail log dari beberapa akun](#)

Setiap entri peristiwa atau log berisi informasi tentang siapa yang membuat permintaan tersebut. Informasi identitas membantu Anda menentukan hal berikut ini:

- Apakah permintaan tersebut dibuat dengan kredensial root atau pengguna IAM.
- Apakah permintaan tersebut dibuat dengan kredensial keamanan sementara untuk satu peran atau pengguna terfederasi.
- Apakah permintaan itu dibuat oleh AWS layanan lain.

Untuk informasi selengkapnya, lihat elemen [CloudTrail UserIdentity](#).

### Data rahasia di AWS CloudTrail

Karena kumpulan pengguna dan kumpulan identitas memproses data pengguna, Amazon Cognito mengaburkan beberapa bidang pribadi di acara Anda CloudTrail dengan nilainya. `HIDDEN_FOR_SECURITY_REASONS` Untuk contoh bidang yang tidak diisi Amazon Cognito ke acara, lihat. [Memahami peristiwa masuk Amazon Cognito](#) Amazon Cognito hanya mengaburkan beberapa bidang yang biasanya berisi informasi pengguna, seperti kata sandi dan token. Amazon Cognito tidak melakukan deteksi otomatis atau penyembunyian informasi identifikasi pribadi yang Anda isi ke bidang non-pribadi dalam permintaan API Anda.

### Kolam Pengguna Amazon Cognito

Amazon Cognito mendukung pencatatan untuk semua tindakan yang tercantum di halaman [tindakan kumpulan Pengguna](#) sebagai peristiwa dalam file CloudTrail log. Amazon Cognito mencatat peristiwa kumpulan pengguna ke CloudTrail acara manajemen.

`eventTypeBidang` dalam CloudTrail entri kumpulan pengguna Amazon Cognito memberi tahu Anda apakah aplikasi Anda membuat permintaan ke API [kumpulan pengguna Amazon Cognito](#) atau ke [titik akhir yang menyajikan sumber daya untuk OpenID Connect, SAFL 2.0](#), atau UI yang dihosting. Permintaan API memiliki permintaan `eventType` of `AwsApiCall` dan titik akhir memiliki `eventType` of `AwsServiceEvent`

Amazon Cognito mencatat permintaan UI yang dihosting berikut ke UI yang dihosting sebagai acara di. CloudTrail

## Operasi UI yang dihosting di CloudTrail

Operasi	Deskripsi
Login_GET , CognitoAuthentication	Pengguna melihat atau mengirimkan kredensya l ke Anda. <a href="#">Titik akhir masuk</a>
OAuth2_Authorize_GET , Beta_Authorize_GET	Seorang pengguna melihat Anda <a href="#">Otorisasi titik akhir</a> .
OAuth2Response_GET , OAuth2Response_POST	Pengguna mengirimkan token iDP ke /oauth2/idpresponse titik akhir Anda.
SAML2Response_POST , Beta_SAML2Response_POST	Seorang pengguna mengirimkan pernyataan IDP SAMP ke titik akhir Anda. /saml2/idpresponse
Login_OIDC_SAML_POST	Pengguna memasukkan nama pengguna di Anda <a href="#">Titik akhir masuk</a> dan cocok dengan pengenal <a href="#">IDP</a> .
Token_POST , Beta_Token_POST	Seorang pengguna mengirimkan kode otorisasi ke Anda. <a href="#">Titik akhir token</a>
Signup_GET , Signup_POST	Pengguna mengirimkan informasi pendaftaran ke titik akhir Anda/signup.
Confirm_GET , Confirm_POST	Pengguna mengirimkan kode konfirmasi di UI yang dihosting.
ResendCode_POST	Pengguna mengirimkan permintaan untuk mengirim ulang kode konfirmasi di UI yang dihosting.
ForgotPassword_GET , ForgotPassword_POST	Pengguna mengirimkan permintaan untuk mengatur ulang kata sandi mereka ke titik /forgotPassword akhir Anda.
ConfirmForgotPassword_GET , ConfirmForgotPassword_POST	Pengguna mengirimkan kode ke /confirmForgotPassword titik akhir Anda yang

Operasi	Deskripsi
	mengonfirmasi permintaan merekaForgotPassword .
ResetPassword_GET , ResetPassword_POST	Pengguna mengirimkan kata sandi baru di UI yang dihosting.
Mfa_GET, Mfa_POST	Pengguna mengirimkan kode otentikasi multi-faktor (MFA) di UI yang dihosting.
MfaOption_GET , MfaOption_POST	Seorang pengguna memilih metode pilihan mereka untuk MFA di UI yang dihosting.
MfaRegister_GET , MfaRegister_POST	Seorang pengguna mengirimkan kode otentikasi multi-faktor (MFA) di UI yang dihosting saat mendaftarkan MFA.
Logout	Seorang pengguna keluar di /logout titik akhir Anda.
SAML2Logout_POST	Seorang pengguna keluar di /saml2/logout titik akhir Anda.
Error_GET	Pengguna melihat halaman kesalahan di UI yang dihosting.
UserInfo_GET , UserInfo_POST	Pengguna atau IDP bertukar informasi dengan Anda. <a href="#">Titik akhir UserInfo</a>
Confirm_With_Link_GET	Pengguna mengirimkan konfirmasi berdasarkan tautan yang dikirim Amazon Cognito dalam pesan email.
Event_Feedback_GET	Pengguna mengirimkan umpan balik ke Amazon Cognito tentang <a href="#">peristiwa fitur keamanan tingkat lanjut</a> .

**Note**

Amazon Cognito mencatat UserSub tetapi tidak UserName dalam CloudTrail log untuk permintaan yang khusus untuk pengguna. Anda dapat menemukan pengguna UserSub dengan memanggil API ListUsers, dan menggunakan filter untuk sub.

## Kumpulan identitas Amazon Cognito

### Peristiwa data

Amazon Cognito mencatat peristiwa Identitas Amazon Cognito berikut sebagai peristiwa data CloudTrail . [Peristiwa data](#) adalah operasi API data-plane volume tinggi yang CloudTrail tidak masuk secara default. Biaya tambahan berlaku untuk peristiwa data.

- [GetCredentialsForIdentity](#)
- [GetId](#)
- [GetOpenIdToken](#)
- [GetOpenIdTokenForDeveloperIdentity](#)
- [UnlinkIdentity](#)

Untuk menghasilkan CloudTrail log untuk operasi API ini, Anda harus mengaktifkan peristiwa data di jejak Anda dan memilih pemilih peristiwa untuk kumpulan identitas Cognito. Untuk informasi lebih lanjut, lihat [Peristiwa Pencatatan Data untuk Pelacakan](#) dalam AWS CloudTrail Panduan Pengguna.

Anda juga dapat menambahkan pemilih acara kumpulan identitas ke jejak Anda dengan perintah CLI berikut.

```
aws cloudtrail put-event-selectors --trail-name <trail name> --advanced-event-selectors
\
"{\
  \"Name\": \"Cognito Selector\", \
  \"FieldSelectors\": [\
    {\
      \"Field\": \"eventCategory\", \
      \"Equals\": [\
        \"Data\" \
      ] \
    } \
  ], \
}
```

```
{\n  \"Field\": \"resources.type\",\n  \"Equals\": [\n    \"AWS::Cognito::IdentityPool\"\n  ]\n}\n]\"
```

## Acara manajemen

Amazon Cognito mencatat sisa operasi API kumpulan identitas Amazon Cognito sebagai peristiwa manajemen. CloudTrail mencatat operasi API peristiwa manajemen log secara default.

Untuk daftar operasi API kumpulan identitas Amazon Cognito yang dicatat oleh Amazon Cognito, lihat Referensi CloudTrail API kumpulan identitas Amazon [Cognito](#).

## Sinkronisasi Amazon Cognito

Amazon Cognito mencatat semua operasi API Sinkronisasi Amazon Cognito sebagai peristiwa manajemen. Untuk daftar operasi API Sinkronisasi Amazon Cognito yang dicatat oleh Amazon Cognito, lihat Referensi API Sinkronisasi Amazon [Cognito](#). CloudTrail

## Memahami peristiwa masuk Amazon Cognito

Jejak dapat mengirimkan peristiwa sebagai file log ke bucket Amazon S3 yang Anda tentukan. CloudTrail file log berisi satu atau lebih entri log. Peristiwa mewakili permintaan tunggal dari sumber mana pun dan mencakup informasi tentang tindakan yang diminta, tanggal dan waktu tindakan, parameter permintaan, dan sebagainya. CloudTrail file log bukanlah jejak tumpukan yang diurutkan dari panggilan API publik, sehingga file tersebut tidak muncul dalam urutan tertentu.

## Topik

- [Contoh CloudTrail peristiwa untuk pendaftaran UI yang dihosting](#)
- [Contoh CloudTrail acara untuk permintaan SAMB](#)
- [Contoh CloudTrail peristiwa untuk permintaan ke titik akhir token](#)
- [Contoh CloudTrail acara untuk CreateIdentityPool](#)
- [Contoh CloudTrail acara untuk GetCredentialsForIdentity](#)
- [Contoh CloudTrail acara untuk GetId](#)
- [Contoh CloudTrail acara untuk GetOpenIdToken](#)

- [Contoh CloudTrail acara untuk GetOpenIdTokenForDeveloperIdentity](#)
- [Contoh CloudTrail acara untuk UnlinkIdentity](#)

Contoh CloudTrail peristiwa untuk pendaftaran UI yang dihosting

Contoh CloudTrail peristiwa berikut menunjukkan informasi yang dicatat oleh Amazon Cognito saat pengguna mendaftar melalui UI yang dihosting.

Amazon Cognito mencatat peristiwa berikut saat pengguna baru menavigasi ke halaman login untuk aplikasi Anda.

```
{
  "eventVersion": "1.08",
  "userIdentity":
  {
    "accountId": "123456789012"
  },
  "eventTime": "2022-04-06T05:38:12Z",
  "eventSource": "cognito-idp.amazonaws.com",
  "eventName": "Login_GET",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "192.0.2.1",
  "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7)...",
  "errorCode": "",
  "errorMessage": "",
  "additionalEventData":
  {
    "responseParameters":
    {
      "status": 200.0
    },
    "requestParameters":
    {
      "redirect_uri":
      [
        "https://www.amazon.com"
      ],
      "response_type":
      [
        "token"
      ],
      "client_id":
```



```
        [
            "1example23456789"
        ]
    },
    "eventID": "382ae09a-151d-4116-8f2b-6ac0a804a38c",
    "readOnly": true,
    "eventType": "AwsServiceEvent",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "serviceEventDetails":
    {
        "serviceAccountId": "111122223333"
    },
    "eventCategory": "Management"
}
```

Amazon Cognito mencatat peristiwa berikut saat pengguna baru memilih Daftar dari halaman login untuk aplikasi Anda.

```
{
    "eventVersion": "1.08",
    "userIdentity":
    {
        "accountId": "123456789012"
    },
    "eventTime": "2022-05-05T23:21:43Z",
    "eventSource": "cognito-idp.amazonaws.com",
    "eventName": "Signup_GET",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.1",
    "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7)...",
    "requestParameters": null,
    "responseElements": null,
    "additionalEventData":
    {
        "responseParameters":
        {
            "status": 200
        },
        "requestParameters":
        {
            "response_type":
```

```

    [
      "code"
    ],
    "redirect_uri":
    [
      "https://www.amazon.com"
    ],
    "client_id":
    [
      "1example23456789"
    ]
  },
  "userPoolDomain": "mydomain.us-west-2.amazoncognito.com",
  "userPoolId": "us-west-2_aaaaaaaaa"
},
"requestID": "7a63e7c2-b057-4f3d-a171-9d9113264fff",
"eventID": "5e7b27a0-6870-4226-adb4-f86cd51ac5d8",
"readOnly": true,
"eventType": "AwsServiceEvent",
"managementEvent": true,
"recipientAccountId": "123456789012",
"serviceEventDetails":
{
  "serviceAccountId": "111122223333"
},
"eventCategory": "Management"
}

```

Amazon Cognito mencatat peristiwa berikut saat pengguna baru memilih nama pengguna, memasukkan alamat email, dan memilih kata sandi dari halaman login untuk aplikasi Anda. Amazon Cognito tidak mencatat informasi identifikasi tentang identitas pengguna. CloudTrail

```

{
  "eventVersion": "1.08",
  "userIdentity":
  {
    "accountId": "123456789012"
  },
  "eventTime": "2022-05-05T23:22:05Z",
  "eventSource": "cognito-idp.amazonaws.com",
  "eventName": "Signup_POST",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "192.0.2.1",

```

```
"userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7)...",
"requestParameters": null,
"responseElements": null,
"additionalEventData":
{
  "responseParameters":
  {
    "status": 302
  },
  "requestParameters":
  {
    "password":
    [
      "HIDDEN_DUE_TO_SECURITY_REASONS"
    ],
    "requiredAttributes[email]":
    [
      "HIDDEN_DUE_TO_SECURITY_REASONS"
    ],
    "response_type":
    [
      "code"
    ],
    "_csrf":
    [
      "HIDDEN_DUE_TO_SECURITY_REASONS"
    ],
    "redirect_uri":
    [
      "https://www.amazon.com"
    ],
    "client_id":
    [
      "1example23456789"
    ],
    "username":
    [
      "HIDDEN_DUE_TO_SECURITY_REASONS"
    ]
  },
  "userPoolDomain": "mydomain.us-west-2.amazoncognito.com",
  "userPoolId": "us-west-2_aaaaaaaaa"
},
"requestID": "9ad58dd8-3517-4aa8-96a5-d17a01df9eb4",
```

```
"eventID": "c75eb7a5-eb8c-43d1-8331-f4412e756e69",
"readOnly": false,
"eventType": "AwsServiceEvent",
"managementEvent": true,
"recipientAccountId": "123456789012",
"serviceEventDetails":
{
  "serviceAccountId": "111122223333"
},
"eventCategory": "Management"
}
```

Amazon Cognito mencatat peristiwa berikut saat pengguna baru mengakses halaman konfirmasi pengguna di UI yang dihosting setelah mereka mendaftar.

```
{
  "eventVersion": "1.08",
  "userIdentity":
  {
    "accountId": "123456789012"
  },
  "eventTime": "2022-05-05T23:22:06Z",
  "eventSource": "cognito-idp.amazonaws.com",
  "eventName": "Confirm_GET",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "192.0.2.1",
  "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7)...",
  "requestParameters": null,
  "responseElements": null,
  "additionalEventData":
  {
    "responseParameters":
    {
      "status": 200
    },
    "requestParameters":
    {
      "response_type":
      [
        "code"
      ],
      "redirect_uri":
      [
```

```

        "https://www.amazon.com"
    ],
    "client_id":
    [
        "1example23456789"
    ]
},
"userPoolDomain": "mydomain.us-west-2.amazoncognito.com",
"userPoolId": "us-west-2_aaaaaaaaa"
},
"requestID": "58a5b170-3127-45bb-88cc-3e652d779e0b",
"eventID": "7f87291a-6d50-409a-822f-e3a5ec7e60da",
"readOnly": false,
"eventType": "AwsServiceEvent",
"managementEvent": true,
"recipientAccountId": "123456789012",
"serviceEventDetails":
{
    "serviceAccountId": "111122223333"
},
"eventCategory": "Management"
}

```

Amazon Cognito mencatat peristiwa berikut ketika, di halaman konfirmasi pengguna di UI yang dihosting, pengguna memasukkan kode yang dikirimkan Amazon Cognito kepada mereka dalam pesan email.

```

{
    "eventVersion": "1.08",
    "userIdentity":
    {
        "accountId": "123456789012"
    },
    "eventTime": "2022-05-05T23:23:32Z",
    "eventSource": "cognito-idp.amazonaws.com",
    "eventName": "Confirm_POST",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.1",
    "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7)...",
    "requestParameters": null,
    "responseElements": null,
    "additionalEventData":
    {

```

```
"responseParameters":
{
  "status": 302
},
"requestParameters":
{
  "confirm":
  [
    ""
  ],
  "deliveryMedium":
  [
    "EMAIL"
  ],
  "sub":
  [
    "704b1e47-34fe-40e9-8c41-504997494531"
  ],
  "code":
  [
    "HIDDEN_DUE_TO_SECURITY_REASONS"
  ],
  "destination":
  [
    "HIDDEN_DUE_TO_SECURITY_REASONS"
  ],
  "response_type":
  [
    "code"
  ],
  "_csrf":
  [
    "HIDDEN_DUE_TO_SECURITY_REASONS"
  ],
  "cognitoAsfData":
  [
    "HIDDEN_DUE_TO_SECURITY_REASONS"
  ],
  "redirect_uri":
  [
    "https://www.amazon.com"
  ],
  "client_id":
  [
```

```

        "1example23456789"
    ],
    "username":
    [
        "HIDDEN_DUE_TO_SECURITY_REASONS"
    ]
},
"userPoolDomain": "mydomain.us-west-2.amazoncognito.com",
"userPoolId": "us-west-2_aaaaaaaaaa"
},
"requestID": "9764300a-ed35-4f87-8a0f-b18b3fe2b11e",
"eventID": "e24ac6e5-2f70-4c6e-ad4e-2f08a547bb36",
"readOnly": false,
"eventType": "AwsServiceEvent",
"managementEvent": true,
"recipientAccountId": "123456789012",
"serviceEventDetails":
{
    "serviceAccountId": "111122223333"
},
"eventCategory": "Management"
}

```

### Contoh CloudTrail acara untuk permintaan SAMB

Amazon Cognito mencatat peristiwa berikut ketika pengguna yang telah mengautentikasi dengan SALL IDP Anda mengirimkan pernyataan SAMP ke titik akhir Anda. `/saml2/idpresponse`

```

{
  "eventVersion": "1.08",
  "userIdentity":
  {
    "accountId": "123456789012"
  },
  "eventTime": "2022-05-06T00:50:57Z",
  "eventSource": "cognito-idp.amazonaws.com",
  "eventName": "SAML2Response_POST",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "192.0.2.1",
  "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7)...",
  "requestParameters": null,
  "responseElements": null,
  "additionalEventData":

```

```

{
  "responseParameters":
  {
    "status": 302
  },
  "requestParameters":
  {
    "RelayState":
    [
      "HIDDEN_DUE_TO_SECURITY_REASONS"
    ],
    "SAMLResponse":
    [
      "HIDDEN_DUE_TO_SECURITY_REASONS"
    ]
  },
  "userPoolDomain": "mydomain.us-west-2.amazoncognito.com",
  "userPoolId": "us-west-2_aaaaaaaaa"
},
"requestID": "4f6f15d1-c370-4a57-87f0-aac4817803f7",
"eventID": "9824b50f-d9d1-4fb8-a2c1-6aa78ca5902a",
"readOnly": false,
"eventType": "AwsServiceEvent",
"managementEvent": true,
"recipientAccountId": "625647942648",
"serviceEventDetails":
{
  "serviceAccountId": "111122223333"
},
"eventCategory": "Management"
}

```

Contoh CloudTrail peristiwa untuk permintaan ke titik akhir token

Berikut ini adalah contoh peristiwa dari permintaan ke [Titik akhir token](#).

Amazon Cognito mencatat peristiwa berikut ketika pengguna yang telah mengautentikasi dan menerima kode otorisasi mengirimkan kode ke titik akhir Anda. /oauth2/token

```

{
  "eventVersion": "1.08",
  "userIdentity":
  {

```



```
    "accountId": "123456789012"
  },
  "eventTime": "2022-05-12T22:12:30Z",
  "eventSource": "cognito-idp.amazonaws.com",
  "eventName": "Token_POST",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "192.0.2.1",
  "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7)...",
  "requestParameters": null,
  "responseElements": null,
  "additionalEventData":
  {
    "responseParameters":
    {
      "status": 200
    },
    "requestParameters":
    {
      "code":
      [
        "HIDDEN_DUE_TO_SECURITY_REASONS"
      ],
      "grant_type":
      [
        "authorization_code"
      ],
      "redirect_uri":
      [
        "https://www.amazon.com"
      ],
      "client_id":
      [
        "1example23456789"
      ]
    },
    "userPoolDomain": "mydomain.us-west-2.amazoncognito.com",
    "userPoolId": "us-west-2_aaaaaaaaa"
  },
  "requestID": "f257f752-cc14-4c52-ad5b-152a46915238",
  "eventID": "0bd1586d-cd3e-4d7a-abaf-fd8bfc3912fd",
  "readOnly": false,
  "eventType": "AwsServiceEvent",
  "managementEvent": true,
  "recipientAccountId": "123456789012",
```

```
"serviceEventDetails":
{
  "serviceAccountId": "111122223333"
},
"eventCategory": "Management"
}
```

Amazon Cognito mencatat peristiwa berikut saat sistem backend Anda mengirimkan `client_credentials` permintaan token akses ke titik akhir Anda. `/oauth2/token`

```
{
  "eventVersion": "1.08",
  "userIdentity":
  {
    "accountId": "123456789012"
  },
  "eventTime": "2022-05-12T21:07:05Z",
  "eventSource": "cognito-idp.amazonaws.com",
  "eventName": "Token_POST",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "192.0.2.1",
  "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7)...",
  "requestParameters": null,
  "responseElements": null,
  "additionalEventData":
  {
    "responseParameters":
    {
      "status": 200
    },
    "requestParameters":
    {
      "grant_type":
      [
        "client_credentials"
      ],
      "client_id":
      [
        "1example23456789"
      ]
    },
    "userPoolDomain": "mydomain.us-west-2.amazoncognito.com",
    "userPoolId": "us-west-2_aaaaaaaa"
  }
}
```

```

    },
    "requestID": "4f871256-6825-488a-871b-c2d9f55caff2",
    "eventID": "473e5cbc-a5b3-4578-9ad6-3dfdc8a6d34",
    "readOnly": false,
    "eventType": "AwsServiceEvent",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "serviceEventDetails":
    {
        "serviceAccountId": "111122223333"
    },
    "eventCategory": "Management"
}

```

Amazon Cognito mencatat peristiwa berikut saat aplikasi Anda menukar token penyegaran dengan ID baru dan token akses dengan titik akhir Anda/oauth2/token.

```

{
  "eventVersion": "1.08",
  "userIdentity":
  {
    "accountId": "123456789012"
  },
  "eventTime": "2022-05-12T22:16:40Z",
  "eventSource": "cognito-idp.amazonaws.com",
  "eventName": "Token_POST",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "192.0.2.1",
  "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7)...",
  "requestParameters": null,
  "responseElements": null,
  "additionalEventData":
  {
    "responseParameters":
    {
      "status": 200
    },
    "requestParameters":
    {
      "refresh_token":
      [
        "HIDDEN_DUE_TO_SECURITY_REASONS"
      ],
    },
  },
}

```

```

    "grant_type":
    [
      "refresh_token"
    ],
    "client_id":
    [
      "1example23456789"
    ]
  },
  "userPoolDomain": "mydomain.us-west-2.amazoncognito.com",
  "userPoolId": "us-west-2_aaaaaaaaa"
},
"requestID": "2829f0c6-a3a9-4584-b046-11756dfe8a81",
"eventID": "12bd3464-59c7-44fa-b8ff-67e1cf092018",
"readOnly": false,
"eventType": "AwsServiceEvent",
"managementEvent": true,
"recipientAccountId": "123456789012",
"serviceEventDetails":
{
  "serviceAccountId": "111122223333"
},
"eventCategory": "Management"
}

```

### Contoh CloudTrail acara untuk CreateIdentityPool

Contoh berikut adalah entri log untuk permintaan tindakan CreateIdentityPool. Permintaan dibuat oleh pengguna IAM bernama Alice.

```

{
  "eventVersion": "1.03",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "PRINCIPAL_ID",
    "arn": "arn:aws:iam::123456789012:user/Alice",
    "accountId": "123456789012",
    "accessKeyId": "['EXAMPLE_KEY_ID']",
    "userName": "Alice"
  },
  "eventTime": "2016-01-07T02:04:30Z",
  "eventSource": "cognito-identity.amazonaws.com",
  "eventName": "CreateIdentityPool",

```

```

"awsRegion": "us-east-1",
"sourceIPAddress": "127.0.0.1",
"userAgent": "USER_AGENT",
"requestParameters": {
  "identityPoolName": "TestPool",
  "allowUnauthenticatedIdentities": true,
  "supportedLoginProviders": {
    "graph.facebook.com": "0000000000000000"
  }
},
"responseElements": {
  "identityPoolName": "TestPool",
  "identityPoolId": "us-east-1:1cf667a2-49a6-454b-9e45-23199EXAMPLE",
  "allowUnauthenticatedIdentities": true,
  "supportedLoginProviders": {
    "graph.facebook.com": "0000000000000000"
  }
},
"requestID": "15cc73a1-0780-460c-91e8-e12ef034e116",
"eventID": "f1d47f93-c708-495b-bff1-cb935a6064b2",
"eventType": "AwsApiCall",
"recipientAccountId": "123456789012"
}

```

## Contoh CloudTrail acara untuk GetCredentialsForIdentity

Contoh berikut adalah entri log untuk permintaan tindakan GetCredentialsForIdentity.

```

{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "Unknown"
  },
  "eventTime": "2023-01-19T16:55:08Z",
  "eventSource": "cognito-identity.amazonaws.com",
  "eventName": "GetCredentialsForIdentity",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "192.0.2.4",
  "userAgent": "aws-cli/2.7.25 Python/3.9.11 Darwin/21.6.0 exe/x86_64 prompt/off
command/cognito-identity.get-credentials-for-identity",
  "requestParameters": {
    "logins": {
      "cognito-idp.us-east-1.amazonaws.com/us-east-1_aaaaaaaaa":
"HIDDEN_DUE_TO_SECURITY_REASONS"

```



```

    "logins": {
      "cognito-idp.us-east-1.amazonaws.com/us-east-1_aaaaaaaaa":
"HIDDEN_DUE_TO_SECURITY_REASONS"
    }
  },
  "responseElements": {
    "identityId": "us-east-1:1cf667a2-49a6-454b-9e45-23199EXAMPLE"
  },
  "requestID": "dc28def9-07c8-460a-a8f3-3816229e6664",
  "eventID": "c5c459d9-40ec-41fd-8f6b-57865d5a9975",
  "readOnly": false,
  "resources": [{
    "accountId": "111122223333",
    "type": "AWS::Cognito::IdentityPool",
    "ARN": "arn:aws:cognito-identity:us-east-1:111122223333:identitypool/us-
east-1:2dg778b3-50b7-565c-0f56-34200EXAMPLE"
  }],
  "eventType": "AwsApiCall",
  "managementEvent": false,
  "recipientAccountId": "111122223333",
  "eventCategory": "Data"
}

```

## Contoh CloudTrail acara untuk GetOpenIdToken

Contoh berikut adalah entri log untuk permintaan tindakan GetOpenIdToken.

```

{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "Unknown"
  },
  "eventTime": "2023-01-19T16:55:08Z",
  "eventSource": "cognito-identity.amazonaws.com",
  "eventName": "GetOpenIdToken",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "192.0.2.4",
  "userAgent": "aws-cli/2.7.25 Python/3.9.11 Darwin/21.6.0 exe/x86_64 prompt/off
command/cognito-identity.get-open-id-token",
  "requestParameters": {
    "identityId": "us-east-1:1cf667a2-49a6-454b-9e45-23199EXAMPLE",
    "logins": {
      "cognito-idp.us-east-1.amazonaws.com/us-east-1_aaaaaaaaa":
"HIDDEN_DUE_TO_SECURITY_REASONS"
    }
  }
}

```

```

    }
  },
  "responseElements": {
    "identityId": "us-east-1:1cf667a2-49a6-454b-9e45-23199EXAMPLE"
  },
  "requestID": "a506ba18-10d7-4fdb-9548-a8187b2e38bb",
  "eventID": "19ffc1a6-6ed8-4580-a4e1-3062c5ce6457",
  "readOnly": false,
  "resources": [{
    "accountId": "111122223333",
    "type": "AWS::Cognito::IdentityPool",
    "ARN": "arn:aws:cognito-identity:us-east-1:111122223333:identitypool/us-east-1:2dg778b3-50b7-565c-0f56-34200EXAMPLE"
  }],
  "eventType": "AwsApiCall",
  "managementEvent": false,
  "recipientAccountId": "111122223333",
  "eventCategory": "Data"
}

```

## Contoh CloudTrail acara untuk GetOpenIdTokenForDeveloperIdentity

Contoh berikut adalah entri log untuk permintaan tindakan `GetOpenIdTokenForDeveloperIdentity`.

```

{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AROAIEXAMPLE:johns-AssumedRoleSession",
    "arn": "arn:aws:sts::111122223333:assumed-role/Admin/johns-AssumedRoleSession",
    "accountId": "111122223333",
    "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AROAIEXAMPLE",
        "arn": "arn:aws:iam::111122223333:role/Admin",
        "accountId": "111122223333",
        "userName": "Admin"
      }
    },
    "attributes": {
      "creationDate": "2023-01-19T16:53:14Z",

```



```

        "mfaAuthenticated": "false"
      }
    }
  },
  "eventTime": "2023-01-19T16:55:08Z",
  "eventSource": "cognito-identity.amazonaws.com",
  "eventName": "GetOpenIdTokenForDeveloperIdentity",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "27.0.3.154",
  "userAgent": "aws-cli/2.7.25 Python/3.9.11 Darwin/21.6.0 exe/x86_64 prompt/off
command/cognito-identity.get-open-id-token-for-developer-identity",
  "requestParameters": {
    "tokenDuration": 900,
    "identityPoolId": "us-east-1:2dg778b3-50b7-565c-0f56-34200EXAMPLE",
    "logins": {
      "JohnsDeveloperProvider": "HIDDEN_DUE_TO_SECURITY_REASONS"
    }
  },
  "responseElements": {
    "identityId": "us-east-1:1cf667a2-49a6-454b-9e45-23199EXAMPLE"
  },
  "requestID": "b807df87-57e7-4dd6-b90c-b06f46a61c21",
  "eventID": "f26fed91-3340-4d70-91ae-cdf555547b76",
  "readOnly": false,
  "resources": [{
    "accountId": "111122223333",
    "type": "AWS::Cognito::IdentityPool",
    "ARN": "arn:aws:cognito-identity:us-east-1:111122223333:identitypool/us-
east-1:2dg778b3-50b7-565c-0f56-34200EXAMPLE"
  }],
  "eventType": "AwsApiCall",
  "managementEvent": false,
  "recipientAccountId": "111122223333",
  "eventCategory": "Data"
}

```

## Contoh CloudTrail acara untuk UnlinkIdentity

Contoh berikut adalah entri log untuk permintaan tindakan UnlinkIdentity.

```

{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "Unknown"
  }
}

```

```

    },
    "eventTime": "2023-01-19T16:55:08Z",
    "eventSource": "cognito-identity.amazonaws.com",
    "eventName": "UnlinkIdentity",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.4",
    "userAgent": "aws-cli/2.7.25 Python/3.9.11 Darwin/21.6.0 exe/x86_64 prompt/off
command/cognito-identity.unlink-identity",
    "requestParameters": {
      "logins": {
        "cognito-idp.us-east-1.amazonaws.com/us-east-1_aaaaaaaa":
"HIDDEN_DUE_TO_SECURITY_REASONS"
      },
      "identityId": "us-east-1:1cf667a2-49a6-454b-9e45-23199EXAMPLE",
      "loginsToRemove": ["cognito-idp.us-east-1.amazonaws.com/us-east-1_aaaaaaaa"]
    },
    "responseElements": null,
    "requestID": "99c2c8e2-9c29-416f-bb17-b650a5cbada9",
    "eventID": "d8e26126-202a-43c2-b458-3f225efaedc7",
    "readOnly": false,
    "resources": [{
      "accountId": "111122223333",
      "type": "AWS::Cognito::IdentityPool",
      "ARN": "arn:aws:cognito-identity:us-east-1:111122223333:identitypool/us-
east-1:2dg778b3-50b7-565c-0f56-34200EXAMPLE"
    }],
    "eventType": "AwsApiCall",
    "managementEvent": false,
    "recipientAccountId": "111122223333",
    "eventCategory": "Data"
  }
}

```

## Menganalisis CloudTrail peristiwa Amazon Cognito dengan Amazon CloudWatch Logs Insights

Anda dapat mencari dan menganalisis CloudTrail peristiwa Amazon Cognito Anda dengan Amazon CloudWatch Logs Insights. Saat Anda mengonfigurasi jejak Anda untuk mengirim peristiwa ke CloudWatch Log, CloudTrail kirimkan hanya peristiwa yang sesuai dengan pengaturan jejak Anda.

Untuk menanyakan atau meneliti CloudTrail peristiwa Amazon Cognito Anda, di CloudTrail konsol, pastikan Anda memilih opsi Pengelolaan peristiwa di pengaturan jejak sehingga Anda dapat memantau operasi manajemen yang dilakukan pada sumber daya Anda AWS . Secara opsional,

Anda dapat memilih opsi Insights events di setelan jejak saat Anda ingin mengidentifikasi kesalahan, aktivitas yang tidak biasa, atau perilaku pengguna yang tidak biasa di akun Anda.

### Contoh kueri Amazon Cognito

Anda dapat menggunakan kueri berikut di CloudWatch konsol Amazon.

#### Pertanyaan umum

Temukan 25 log acara yang paling baru ditambahkan.

```
fields @timestamp, @message | sort @timestamp desc | limit 25
| filter eventSource = "cognito-idp.amazonaws.com"
```

Dapatkan daftar 25 peristiwa log yang paling baru ditambahkan yang menyertakan pengecualian.

```
fields @timestamp, @message | sort @timestamp desc | limit 25
| filter eventSource = "cognito-idp.amazonaws.com" and @message like /Exception/
```

#### Kueri Pengecualian dan Kesalahan

Menemukan 25 log acara yang paling baru ditambahkan dengan kode kesalahan `NotAuthorizedException` bersama dengan kolam pengguna Amazon Cognito sub.

```
fields @timestamp, additionalEventData.sub as user | sort @timestamp desc | limit 25
| filter eventSource = "cognito-idp.amazonaws.com" and errorCode=
  "NotAuthorizedException"
```

Menemukan jumlah catatan dengan `sourceIPAddress` dan sesuai `eventName`.

```
filter eventSource = "cognito-idp.amazonaws.com"
| stats count(*) by sourceIPAddress, eventName
```

Menemukan 25 alamat IP teratas yang memicu kesalahan `NotAuthorizedException`.

```
filter eventSource = "cognito-idp.amazonaws.com" and errorCode=
  "NotAuthorizedException"
| stats count(*) as count by sourceIPAddress, eventName
| sort count desc | limit 25
```

Menemukan 25 alamat IP teratas yang memanggil API ForgotPassword.

```
filter eventSource = "cognito-idp.amazonaws.com" and eventName = 'ForgotPassword'  
| stats count(*) as count by sourceIPAddress  
| sort count desc | limit 25
```

## Validasi kepatuhan untuk Amazon Cognito

Auditor pihak ketiga menilai keamanan dan kepatuhan Amazon Cognito sebagai bagian dari AWS beberapa program kepatuhan. Program ini mencakup SOC, PCI, FedRAMP, HIPAA, dan lainnya.

Untuk daftar AWS layanan dalam lingkup program kepatuhan tertentu, lihat [AWS layanan dalam lingkup oleh AWS layanan program kepatuhan](#) . Untuk informasi umum, lihat [AWS program kepatuhan](#).

Anda dapat mengunduh laporan audit pihak ketiga menggunakan AWS Artifact. Untuk informasi selengkapnya, lihat [Mengunduh laporan di AWS Artifact](#) .

Tanggung jawab kepatuhan Anda ketika menggunakan Amazon Cognito ditentukan oleh sensitivitas data Anda, tujuan kepatuhan perusahaan Anda, serta peraturan perundang-undangan yang berlaku. AWS menyediakan sumber daya berikut untuk memberi bantuan terkait kepatuhan:

- [Panduan mulai cepat keamanan dan kepatuhan](#) – Panduan deployment ini membahas pertimbangan arsitektural dan menyediakan langkah-langkah untuk men-deploy lingkungan dasar yang berfokus pada keamanan dan kepatuhan pada AWS.
- [Arsitektur untuk whitepaper keamanan dan kepatuhan HIPAA - Whitepaper](#) ini menjelaskan bagaimana perusahaan dapat menggunakan untuk membuat aplikasi yang sesuai dengan HIPAA. AWS
- [AWS sumber daya AWS kepatuhan](#) — Kumpulan buku kerja dan panduan ini mungkin berlaku untuk industri dan lokasi Anda.
- [Mengevaluasi sumber daya dengan aturan](#) dalam Panduan AWS Config Pengembang — AWS Config; menilai seberapa baik konfigurasi sumber daya Anda mematuhi praktik internal, pedoman industri, dan peraturan.
- [AWS Security Hub](#)— AWS Layanan ini memberikan pandangan komprehensif tentang keadaan keamanan Anda di dalamnya AWS yang membantu Anda memeriksa kepatuhan Anda terhadap standar industri keamanan dan praktik terbaik.

# Ketahanan di Amazon Cognito

Infrastruktur AWS global dibangun di sekitar AWS Wilayah dan Zona Ketersediaan. Wilayah memberikan beberapa Zona Ketersediaan yang terpisah dan terisolasi secara fisik, yang terkoneksi melalui jaringan latensi rendah, throughput tinggi, dan sangat redundan. Dengan Zona Ketersediaan, Anda dapat merancang serta mengoperasikan aplikasi dan basis data yang secara otomatis melakukan fail over di antara zona tanpa gangguan. Zona Ketersediaan memiliki ketersediaan dan toleransi kesalahan yang lebih baik, dan dapat diskalakan dibandingkan infrastruktur pusat data tunggal atau multi tradisional.

Untuk informasi selengkapnya tentang AWS Wilayah dan Availability Zone, lihat [infrastruktur AWS global](#).

Topik

- [Pertimbangan data regional](#)

## Pertimbangan data regional

Kumpulan pengguna Amazon Cognito masing-masing dibuat dalam satu AWS Wilayah, dan mereka menyimpan data profil pengguna hanya di wilayah itu. Kumpulan pengguna dapat mengirim data pengguna ke AWS Wilayah yang berbeda, tergantung pada bagaimana fitur opsional dikonfigurasi.

- Jika pengaturan alamat `no-reply@verificationemail.com` email default digunakan untuk merutekan verifikasi alamat email dengan kumpulan pengguna Amazon Cognito, email dirutekan melalui wilayah yang sama dengan kumpulan pengguna terkait.
- Jika alamat email lain digunakan untuk mengonfigurasi Layanan Email Sederhana Amazon (Amazon SES) dengan kumpulan pengguna Amazon Cognito, alamat email tersebut dirutekan AWS melalui Wilayah yang terkait dengan alamat email di Amazon SES.
- Pesan SMS dari kumpulan pengguna Amazon Cognito dirutekan melalui wilayah yang sama Amazon SNS kecuali disebutkan lain pada [Mengonfigurasi](#) verifikasi email atau telepon.
- Jika analitik Amazon Pinpoint digunakan dengan kolam pengguna Amazon Cognito, data peristiwa diarahkan ke Wilayah US East (N. Virginia).

**Note**

Amazon Pinpoint tersedia di beberapa AWS Wilayah di Amerika Utara, Eropa, Asia, dan Oseania. Wilayah Amazon Pinpoint termasuk API Amazon Pinpoint. Jika wilayah Amazon Pinpoint didukung oleh Amazon Cognito, maka Amazon Cognito akan mengirim peristiwa ke proyek Amazon Pinpoint dalam wilayah Amazon Pinpoint yang sama. Jika sebuah wilayah tidak didukung oleh Amazon Pinpoint, maka Amazon Cognito hanya akan mendukung mengirim peristiwa di us-east-1. Untuk informasi wilayah terperinci Amazon Pinpoint, lihat titik akhir dan [kuota Amazon Pinpoint dan Menggunakan analitik Amazon Pinpoint dengan kumpulan pengguna amazon cognito](#).

## Keamanan infrastruktur di Amazon Cognito

Sebagai layanan terkelola, Amazon Cognito dilindungi oleh keamanan jaringan AWS global. Untuk informasi tentang layanan AWS keamanan dan cara AWS melindungi infrastruktur, lihat [Keamanan AWS Cloud](#). Untuk mendesain AWS lingkungan Anda menggunakan praktik terbaik untuk keamanan infrastruktur, lihat [Perlindungan Infrastruktur dalam Kerangka Kerja](#) yang AWS Diarsiteksikan dengan Baik Pilar Keamanan.

Anda menggunakan panggilan API yang AWS dipublikasikan untuk mengakses Amazon Cognito melalui jaringan. Klien harus mendukung hal-hal berikut:

- Keamanan Lapisan Pengangkutan (TLS). Kami mensyaratkan TLS 1.2 dan menganjurkan TLS 1.3.
- Sandi cocok dengan sistem kerahasiaan maju sempurna (perfect forward secrecy, PFS) seperti DHE (Ephemeral Diffie-Hellman) atau ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Sebagian besar sistem modern seperti Java 7 dan versi lebih baru mendukung mode-mode ini.

Selain itu, permintaan harus ditandatangani menggunakan ID kunci akses dan kunci akses rahasia yang terkait dengan prinsipal IAM. Atau Anda dapat menggunakan [AWS Security Token Service](#) (AWS STS) untuk menghasilkan kredensial keamanan sementara untuk menandatangani permintaan.

# Analisis konfigurasi dan kerentanan di kumpulan pengguna Amazon Cognito

AWS menangani tugas-tugas keamanan dasar seperti sistem operasi tamu (OS) dan patch database, konfigurasi firewall, dan pemulihan bencana. Prosedur ini telah ditinjau dan disertifikasi oleh pihak ketiga yang sesuai. Untuk detail selengkapnya, lihat sumber daya berikut:

- [Validasi kepatuhan untuk Amazon Cognito](#)
- [Model Tanggung Jawab Bersama](#)

## AWS kebijakan terkelola untuk Amazon Cognito

Untuk menambahkan izin ke pengguna, grup, dan peran, lebih mudah menggunakan kebijakan AWS terkelola daripada menulis kebijakan sendiri. Dibutuhkan waktu dan keahlian untuk [membuat kebijakan yang dikelola pelanggan IAM](#) yang hanya memberi tim Anda izin yang mereka butuhkan. Untuk memulai dengan cepat, Anda dapat menggunakan kebijakan AWS terkelola kami. Kebijakan ini mencakup kasus penggunaan umum dan tersedia di AWS akun Anda. Untuk informasi selengkapnya tentang kebijakan AWS [AWS terkelola, lihat kebijakan terkelola](#) di Panduan Pengguna IAM.

AWS layanan memelihara dan memperbarui kebijakan AWS terkelola. Anda tidak dapat mengubah izin dalam kebijakan AWS terkelola. Layanan terkadang menambahkan izin tambahan ke kebijakan yang dikelola AWS untuk mendukung fitur-fitur baru. Jenis pembaruan ini akan memengaruhi semua identitas (pengguna, grup, dan peran) di mana kebijakan tersebut dilampirkan. Layanan kemungkinan besar akan memperbarui kebijakan yang dikelola AWS saat ada fitur baru yang diluncurkan atau saat ada operasi baru yang tersedia. Layanan tidak menghapus izin dari kebijakan AWS terkelola, sehingga pembaruan kebijakan tidak akan merusak izin yang ada.

Selain itu, AWS mendukung kebijakan terkelola untuk fungsi pekerjaan yang mencakup beberapa layanan. Misalnya, kebijakan ReadOnlyAccess AWS terkelola menyediakan akses hanya-baca ke semua AWS layanan dan sumber daya. Saat layanan meluncurkan fitur baru, AWS menambahkan izin hanya-baca untuk operasi dan sumber daya baru. Untuk melihat daftar dan deskripsi dari kebijakan fungsi tugas, lihat [kebijakan yang dikelola AWS untuk fungsi tugas](#) di Panduan Pengguna IAM.

Sejumlah kebijakan tersedia melalui Konsol IAM yang dapat Anda gunakan untuk memberikan akses ke Amazon Cognito:

- `AmazonCognitoPowerUser` - Izin untuk mengakses dan mengelola semua aspek kolam identitas dan kolam pengguna Anda. Untuk melihat izin kebijakan ini, lihat [AmazonCognitoPowerUser](#).
- `AmazonCognitoReadOnly` - Izin untuk akses hanya-baca ke kolam identitas dan kolam pengguna Anda. Untuk melihat izin kebijakan ini, lihat [AmazonCognitoReadOnly](#).
- `AmazonCognitoDeveloperAuthenticatedIdentities` - Izin untuk sistem autentikasi Anda untuk diintegrasikan dengan Amazon Cognito. Untuk melihat izin kebijakan ini, lihat [AmazonCognitoDeveloperAuthenticatedIdentities](#).

Kebijakan ini dikelola oleh tim Amazon Cognito, jadi meskipun API baru ditambahkan, pengguna Anda tetap memiliki tingkat akses yang sama.

#### Note

Saat membuat kumpulan identitas baru, Anda dapat secara otomatis membuat peran baru untuk akses pengguna yang diautentikasi dan tamu. Administrator yang membuat kumpulan identitas Anda dengan peran IAM baru juga harus memiliki izin IAM untuk membuat peran.

Kumpulan identitas dengan akses tamu yang tidak diautentikasi menerapkan kebijakan AWS terkelola tambahan `AmazonCognitoUnAuthedIdentitiesSessionPolicy`, sebagai [kebijakan sesi](#) untuk pengguna yang tidak diautentikasi. Kebijakan AWS terkelola ini tidak memiliki penggunaan administratif yang dimaksudkan. Sebaliknya, ini membatasi ruang lingkup izin yang dapat Anda terapkan untuk pengguna tamu di kumpulan identitas alur [otentikasi yang ditingkatkan](#). Untuk informasi selengkapnya, lihat [Peran IAM](#).

## Amazon Cognito memperbarui kebijakan terkelola AWS

Lihat detail tentang pembaruan kebijakan AWS terkelola untuk Amazon Cognito sejak layanan ini mulai melacak perubahan ini. Untuk peringatan otomatis tentang perubahan pada halaman ini, berlangganan umpan RSS di halaman riwayat Dokumen Amazon [Cognito](#).



Perubahan	Deskripsi	Tanggal
AmazonCognitoUnAuthenticatedIdentitiesSessionPolicy —kebijakan baru	Menambahkan kebijakan AWS terkelola untuk cakupan hak istimewa pengguna tamu di kumpulan identitas.	14 Juli 2023
AmazonCognitoPowerUser dan AmazonCognitoReadOnly — Pembaruan kebijakan yang ada	Menambahkan izin baru untuk memungkinkan pengguna daya melihat dan mengelola asosiasi ACL AWS WAF web ke kumpulan pengguna Amazon Cognito.  Menambahkan izin baru untuk memungkinkan pengguna hanya-baca melihat asosiasi ACL AWS WAF web ke kumpulan pengguna Amazon Cognito.	19 Juli 2022
AmazonCognitoPowerUser – Pembaruan ke kebijakan yang ada	Menambahkan izin baru untuk mengizinkan Amazon Cognito memanggil Layanan Email Sederhana Amazon PutIdentityPolicy dan ListConfigurationSets operasi.  Perubahan ini memungkinkan kumpulan pengguna Amazon Cognito memperbarui kebijakan otorisasi pengiriman Amazon SES dan menerapkan set konfigurasi Amazon SES saat Anda mengonfigurasi	17 November 2021

Perubahan	Deskripsi	Tanggal
	pengiriman email di kumpulan pengguna.	
AmazonCognitoPowerUser — Perbaruan ke kebijakan yang sudah ada	<p>Menambahkan izin baru untuk memungkinkan Amazon Cognito memanggil operasi <code>GetSMSSandboxAccountStatus</code> Amazon Simple Notification Service.</p> <p>Perubahan ini memungkinkan kumpulan pengguna Amazon Cognito untuk memutuskan apakah Anda perlu lulus dari kotak pasir Amazon Simple Notification Service untuk mengirim pesan ke semua pengguna akhir melalui kumpulan pengguna.</p>	1 Juni 2021
Amazon Cognito mulai melacak perubahan	Amazon Cognito mulai melacak perubahan untuk kebijakan yang AWS dikelola.	1 Maret 2021

# Menandai sumber daya Amazon Cognito

SEBUAHmenandaiadalah label metadata yang Anda atauAWSmenugaskan keAWSsumber daya. Setiap tanda terdiri dari kunci dan nilai. Untuk tanda yang Anda tetapkan, Anda menentukan kunci dan nilai. Misalnya, Anda dapat menentukan kunci sebagai `stage` dan nilai untuk satu sumber daya sebagai `test`.

Tanda membantu Anda melakukan hal berikut:

- Identifikasi dan organisir sumber daya AWS Anda. BanyakAWSlayanan mendukung penandaan, sehingga Anda dapat menetapkan tag yang sama ke sumber daya dari layanan yang berbeda. Ini membantu Anda menunjukkan sumber daya mana yang terkait. Misalnya, Anda dapat menugaskan tanda yang sama ke kolam pengguna Amazon Cognito yang Anda tetapkan ke tabel Amazon DynamoDB.
- Telusuri biaya AWS Anda. Anda dapat mengaktifkan tag ini diAWS Billing and Cost Managementdasbor.AWSmenggunakan tag alokasi biaya untuk mengkategorikan biaya Anda dan mengirimkan laporan alokasi biaya bulanan kepada Anda. Untuk informasi lebih lanjut, lihat[Gunakan tag alokasi biaya](#)diAWS BillingPanduan Pengguna.
- Kontrol akses ke sumber daya Anda berdasarkan tanda yang ditetapkan untuk sumber daya tersebut. Anda dapat mengontrol akses dengan menentukan kunci tag dan nilai dalam kondisi untukAWS Identity and Access Management(IAM) kebijakan. Misalnya, Anda dapat mengizinkan pengguna untuk memperbarui kumpulan pengguna hanya jika kumpulan pengguna memilikiownerntag dengan nilai nama pengguna itu. Untuk informasi lebih lanjut, lihat[Mengontrol akses menggunakan tag](#)diPanduan Pengguna IAM.

Anda dapat menggunakanAWS Command Line Interfaceatau Amazon Cognito API untuk menambahkan, mengedit, atau menghapus tag untuk kumpulan pengguna dan identitas. Anda juga dapat mengelola tag untuk kumpulan pengguna dengan menggunakan konsol Amazon Cognito.

Untuk tips menggunakan tag, lihat[AWSstrategi penandaan](#)posting diAWSJawabanblog.

Bagian berikut menyediakan informasi selengkapnya tentang tanda untuk Amazon Cognito.

## Sumber daya yang didukung di Amazon Cognito

Sumber daya berikut di Amazon Cognito mendukung penandaan:

- Kolam pengguna
- Kolam identitas

## Pembatasan tanda

Pembatasan berikut berlaku untuk tag di sumber daya Amazon Cognito:

- Jumlah tanda maksimum yang dapat Anda tetapkan ke sumber daya — 50
- Panjang kunci maksimum – 128 karakter Unicode
- Panjang nilai maksimum – 256 karakter Unicode
- Karakter yang valid untuk kunci dan nilai — a-z, A-Z, 0-9, spasi, dan karakter berikut: `_.:/= + - @`
- Kunci dan nilai peka huruf besar dan kecil
- Jangan gunakan `aws :` sebagai prefiks untuk kunci; ini dicadangkan untuk penggunaan AWS

## Mengelola tag menggunakan konsol Amazon Cognito

Anda dapat menggunakan konsol Amazon Cognito untuk mengelola tanda yang ditetapkan ke kolam pengguna Anda.

Untuk menambahkan tanda ke kolam pengguna

1. Arahkan ke [Konsol Amazon Cognito](#). Jika diminta, masukkan AWS kredensialnya.
2. Pilih Kolam Pengguna.
3. Pilih kolam pengguna yang ada dari daftar, atau [buat kolam pengguna](#).
4. Pilih Properti kolam pengguna tab dan temukan Tag.
5. Pilih Tambahkan tag untuk menambahkan tag pertama Anda. Jika sebelumnya Anda telah menetapkan tag ke kumpulan pengguna ini, di [Mengelola tag](#), pilih [Tambahkan](#) yang lain.
6. Tentukan nilai untuk Kunci Tanda dan Nilai Tanda.
7. Untuk setiap tag tambahan yang ingin Anda tambahkan, pilih [Tambahkan](#) yang lain.
8. Setelah Anda selesai menambahkan tanda, pilih [Simpan perubahan](#).

Pada [Mengelola tag](#) halaman, Anda juga dapat mengedit kunci dan nilai dari setiap tag yang ada. Untuk menghapus sebuah tag, pilih [Hapus](#).

## Contoh AWS CLI

The AWS CLI menyediakan perintah yang membantu Anda mengelola tag yang Anda tetapkan ke kumpulan pengguna dan kumpulan identitas Amazon Cognito Anda.

### Menetapkan tanda

Gunakan perintah berikut untuk menetapkan tanda ke kolom pengguna dan kolom identitas yang ada.

Example **tag-resource** Perintah untuk kumpulan pengguna

Menetapkan tanda ke kolom pengguna dengan menggunakan [tag-resource](#) dalam set perintah `cognito-idp`:

```
$ aws cognito-idp tag-resource \  
> --resource-arn user-pool-arn \  
> --tags Stage=Test
```

Perintah ini mencakup parameter-parameter berikut ini:

- `resource-arn` — Amazon Resource Name (ARN) dari kolom pengguna tempat Anda menerapkan tanda. Untuk mencari ARN, pilih kolom pengguna di konsol Amazon Cognito, dan lihat nilai ARN Kolam pada tab Pengaturan umum.
- `tags` — Pasangan kunci-nilai tag, dalam format `key=value`.

Untuk menetapkan beberapa tanda sekaligus, tentukan tanda tersebut dalam daftar yang dipisahkan koma:

```
$ aws cognito-idp tag-resource \  
> --resource-arn user-pool-arn \  
> --tags Stage=Test, CostCenter=80432, Owner=SysEng
```

Example **tag-resource** Perintah untuk kumpulan identitas

Menetapkan tanda ke kolom identitas dengan menggunakan [tag-resource](#) dalam set perintah `cognito-identity`:

```
$ aws cognito-identity tag-resource \  
>
```

```
> --resource-arn identity-pool-arn \  
> --tags Stage=Test
```

Perintah ini mencakup parameter-parameter berikut ini:

- `resource-arn` — Amazon Resource Name (ARN) dari kolom identitas tempat Anda menerapkan tanda. Untuk mencari ARN, pilih kolom identitas di konsol Amazon Cognito, dan pilih Edit kolom identitas. Kemudian, di ID kolom identitas, pilih Tampilkan ARN.
- `tags`— Pasangan kunci-nilai tag, dalam format `key=value`.

Untuk menetapkan beberapa tanda sekaligus, tentukan tanda tersebut dalam daftar yang dipisahkan koma:

```
$ aws cognito-identity tag-resource \  
> --resource-arn identity-pool-arn \  
> --tags Stage=Test, CostCenter=80432, Owner=SysEng
```

## Melihat tanda

Gunakan perintah berikut untuk melihat tanda yang telah Anda tetapkan ke kolom pengguna dan kolom identitas Anda.

Example **list-tags-for-resource** Perintah untuk kumpulan pengguna

Melihat tanda yang ditetapkan ke kolom pengguna dengan menggunakan [list-tags-for-resource](#) dalam set perintah `cognito-idp`:

```
$ aws cognito-idp list-tags-for-resource --resource-arn user-pool-arn
```

Example **list-tags-for-resource** Perintah untuk kumpulan identitas

Melihat tanda yang ditetapkan ke kolom identitas dengan menggunakan [list-tags-for-resource](#) dalam set perintah `cognito-identity`:

```
$ aws cognito-identity list-tags-for-resource --resource-arn identity-pool-arn
```

## Menghapus tanda

Gunakan perintah berikut untuk menghapus tanda dari kolom pengguna dan kolom identitas Anda.

## Example **untag-resource**Perintah untuk kumpulan pengguna

Menghapus tanda dari kolam pengguna dengan menggunakan [untag-resource](#) dalam set perintah `cognito-idp`:

```
$ aws cognito-idp untag-resource \  
> --resource-arn user-pool-arn \  
> --tag-keys Stage CostCenter Owner
```

Untuk `--tag-keys` parameter, tentukan satu atau lebih kunci tag. Jangan sertakan nilai tag. Pisahkan kunci dengan spasi.

## Example **untag-resource**Perintah untuk kumpulan identitas

Menghapus tanda dari kolam identitas dengan menggunakan [untag-resource](#) dalam set perintah `cognito-identity`:

```
$ aws cognito-identity untag-resource \  
> --resource-arn identity-pool-arn \  
> --tag-keys Stage CostCenter Owner
```

Untuk `--tag-keys` parameter, tentukan satu atau lebih kunci tag. Jangan sertakan nilai tag.

### Important

Setelah menghapus pengguna atau kumpulan identitas, tag yang terkait dengan kumpulan yang dihapus masih dapat muncul di konsol atau panggilan API hingga 30 hari setelah penghapusan.

## Menerapkan tanda saat Anda membuat sumber daya

Gunakan perintah berikut untuk menetapkan tanda saat Anda membuat kolam pengguna atau kolam identitas.

### Example **create-user-pool**Perintah dengan tag

Ketika Anda membuat kolam pengguna dengan menggunakan perintah [create-user-pool](#), Anda dapat menentukan tanda dengan parameter `--user-pool-tags`:

```
$ aws cognito-idp create-user-pool \  
> --user-pool-tags Stage CostCenter Owner
```

```
> --pool-name user-pool-name \  
> --user-pool-tags Stage=Test, CostCenter=80432, Owner=SysEng
```

Pasangan nilai kunci untuk tag harus dalam format *key=value*. Jika Anda menambahkan beberapa tag, tentukan dalam daftar yang dipisahkan koma.

Example **create-identity-pool** Perintah dengan tag

Ketika Anda membuat kolam identitas dengan menggunakan perintah [create-identity-pool](#), Anda dapat menentukan tanda dengan parameter `--identity-pool-tags`:

```
$ aws cognito-identity create-identity-pool \  
> --identity-pool-name identity-pool-name \  
> --allow-unauthenticated-identities \  
> --identity-pool-tags Stage=Test, CostCenter=80432, Owner=SysEng
```

Pasangan nilai kunci untuk tag harus dalam format *key=value*. Jika Anda menambahkan beberapa tag, tentukan dalam daftar yang dipisahkan koma.

## Mengelola tag menggunakan Amazon Cognito API

Anda dapat menggunakan tindakan berikut di API Amazon Cognito untuk mengelola tanda untuk kolam pengguna dan kolam identitas Anda.

### Tindakan API untuk tag kumpulan pengguna

Gunakan tindakan API berikut untuk menetapkan, melihat, dan menghapus tanda untuk kolam pengguna.

- [TagResource](#)
- [ListTagsForResource](#)
- [UntagResource](#)
- [CreateUserPool](#)

### Tindakan API untuk tag kumpulan identitas

Gunakan tindakan API berikut untuk menetapkan, melihat, dan menghapus tanda untuk kolam identitas.



- [TagResource](#)
- [ListTagsForResource](#)
- [UntagResource](#)
- [CreateIdentityPool](#)

# Kuota di Amazon Cognito

Amazon Cognito memiliki kuota default, sebelumnya disebut sebagai batas, untuk jumlah maksimum operasi yang dapat Anda lakukan di akun Anda. Amazon Cognito juga memiliki kuota untuk jumlah dan ukuran maksimum sumber daya Amazon Cognito.

Setiap kuota Amazon Cognito mewakili volume maksimum permintaan dalam satu banding satu Wilayah AWS . Akun AWS Misalnya, aplikasi Anda dapat membuat permintaan API hingga tingkat kuota Default (RPS) untuk `UserAuthentication` operasi terhadap semua kumpulan pengguna Anda di US East (Virginia N.). Aplikasi Anda di Asia Pasifik (Tokyo) dapat menghasilkan volume permintaan yang sama terhadap semua kumpulan pengguna Anda di Wilayah mereka sendiri. AWS hanya dapat memberikan permintaan peningkatan kuota di satu Wilayah pada satu waktu. Peningkatan kuota yang berhasil di AS Timur (Virginia N.) tidak berpengaruh pada tingkat permintaan maksimum Anda di Asia Pasifik (Tokyo).

Topik

- [Memahami kuota tingkat permintaan API](#)
- [Mengelola kuota tingkat permintaan API](#)
- [Pengguna Amazon Cognito mengumpulkan kategori operasi API dan kuota tingkat permintaan](#)
- [Kumpulan identitas Amazon Cognito \(identitas federasi\) kuota tingkat permintaan operasi API](#)
- [Kuota pada jumlah dan ukuran sumber daya](#)

## Memahami kuota tingkat permintaan API

### Kategorisasi kuota

Amazon Cognito memberlakukan tingkat permintaan maksimum untuk operasi API. Untuk informasi selengkapnya tentang operasi API yang disediakan Amazon Cognito, lihat [API Amazon Cognito dan referensi titik akhir](#) Untuk kumpulan pengguna, operasi ini dikelompokkan ke dalam kategori kasus penggunaan umum seperti atau. `UserAuthentication` `UserCreation` Untuk daftar operasi API kumpulan pengguna berdasarkan kategori, lihat [Pengguna Amazon Cognito mengumpulkan kategori operasi API dan kuota tingkat permintaan](#).

Di [konsol Service Quotas](#), Anda dapat melacak penggunaan kuota berdasarkan kumpulan pengguna kategori dan kumpulan identitas. Jika tingkat permintaan kumpulan pengguna Amazon Cognito Anda

atau melebihi kuota, Anda dapat membeli kapasitas tambahan. Anda dapat melacak penggunaan kuota kumpulan pengguna berdasarkan kategori dan peningkatan kuota pembelian di konsol [Service Quotas](#).

Kuota operasi didefinisikan sebagai jumlah maksimum permintaan per detik (RPS) untuk semua operasi dalam suatu kategori. Layanan kolam pengguna Amazon Cognito menerapkan kuota untuk semua operasi di setiap kategori. Misalnya, kategori tersebut `UserCreation` mencakup empat operasi: `SignUp`, `ConfirmSignUp`, `AdminCreateUser`, dan `AdminConfirmSignUp`. Ini dialokasikan dengan kuota gabungan 50 RPS. Jika beberapa operasi berlangsung pada saat yang sama, setiap operasi dalam kategori ini dapat memanggil hingga 50 RPS secara terpisah atau digabungkan.

#### Note

Kuota kategori hanya berlaku untuk kumpulan pengguna. Amazon Cognito menerapkan setiap kuota kumpulan identitas ke satu operasi. Untuk kuota tingkat permintaan per kategori dan per operasi, AWS ukur tingkat agregat semua permintaan dari semua kumpulan pengguna atau kumpulan identitas di satu Wilayah Anda. Akun AWS

## Pengguna Amazon Cognito menggabungkan operasi API dengan penanganan tingkat permintaan khusus

Kuota operasi diukur dan diberlakukan untuk total permintaan gabungan pada tingkat kategori, kecuali untuk `AdminRespondToAuthChallenge` dan `RespondToAuthChallenge` operasi, di mana aturan penanganan khusus diterapkan.

`UserAuthentication` kategori ini mencakup empat operasi di API kumpulan pengguna Amazon Cognito: `AdminInitiateAuth`, `InitiateAuthAdminRespondToAuthChallenge`, dan `RespondToAuthChallenge`. Selain itu, otentikasi pengguna di UI yang dihosting berkontribusi pada kuota ini. Operasi `InitiateAuth` dan `AdminInitiateAuth` diukur dan diberlakukan per kuota kategori. Operasi pencocokan `RespondToAuthChallenge` dan `AdminRespondToAuthChallenge` tunduk pada kuota terpisah yang tiga kali batas `UserAuthentication` kategori. Kuota yang ditinggikan ini mengakomodasi beberapa tantangan autentikasi yang disiapkan di aplikasi Anda. Kuota cukup untuk menutupi sebagian besar kasus penggunaan. Setelah aplikasi Anda membuat hingga tiga respons terhadap tantangan autentikasi, permintaan tambahan dihitung dalam kuota `UserAuthentication` kategori. [Otentikasi multi-faktor \(MFA\)](#), otentikasi [perangkat](#), dan [otentikasi kustom](#) adalah contoh permintaan tantangan yang mungkin Anda rekayasa ke dalam kumpulan pengguna Anda.

Misalnya, jika kuota Anda untuk `UserAuthentication` kategori adalah 80 RPS, Anda dapat menelepon `RespondToAuthChallenge` atau dengan `AdminRespondToAuthChallenge` tarif hingga 240 RPS ( $3 * 80$  RPS). Jika kumpulan pengguna Anda meminta empat putaran tantangan per otentikasi dan 70 pengguna masuk per detik, maka totalnya `RespondToAuthChallenge` adalah 280 RPS ( $70 * 4$ ), yaitu 40 RPS di atas kuota. Tambahan 40 RPS ditambahkan ke 70 `InitiateAuth` panggilan, membuat total penggunaan `UserAuthentication` kategori 110 RPS ( $40 + 70$ ). Karena nilai ini melebihi kuota kategori yang ditetapkan pada 80 RPS oleh 30 RPS, Amazon Cognito membatasi permintaan dari aplikasi Anda.

## Pengguna aktif bulanan

Saat Amazon Cognito menghitung penagihan kumpulan pengguna, Amazon Cognito membebankan tarif untuk setiap pengguna aktif bulanan (MAU). Pertimbangkan jumlah MAU Anda saat ini dan yang diproyeksikan dalam perencanaan Anda untuk permintaan peningkatan kuota. Seorang pengguna dihitung sebagai MAU jika, dalam satu bulan kalender, ada operasi identitas yang terkait dengan pengguna tersebut. Aktivitas yang membuat pengguna aktif termasuk yang berikut ini.

- Pendaftaran atau pembuatan administratif pengguna
- Masuk
- Keluar
- Konfirmasi akun pengguna atau verifikasi atribut
- Pengaturan ulang kata sandi
- Ubah atribut pengguna, keanggotaan grup, atau preferensi MFA
- Kueri atribut rinci dari pengguna
- Aktivasi pengguna, penonaktifan atau penghapusan

### Note

Kategori Query atribut rinci pengguna menyertakan operasi API [AdminGetUser](#), tetapi tidak [ListUsers](#). user-by-user Kueri terperinci di kumpulan pengguna yang besar dapat berdampak signifikan pada AWS tagihan Anda. Untuk menghindari biaya berlebih, kumpulkan data pengguna dengan `ListUsers` atau simpan informasi pengguna dalam database eksternal.

# Mengelola kuota tingkat permintaan API

## Mengidentifikasi persyaratan kuota

### Important

Jika Anda meningkatkan kuota Amazon Cognito untuk kategori seperti `UserAuthentication`, atau `UserCreationAccountRecovery`, Anda mungkin perlu menambah kuota untuk yang lain. Layanan AWS Misalnya, pesan yang dikirim Amazon Cognito dengan Amazon Simple Notification Service (Amazon SNS) atau Amazon Simple Email Service (Amazon SES) dapat gagal jika kuota tingkat permintaan tidak mencukupi dalam layanan tersebut.

Untuk menghitung persyaratan kuota, tentukan berapa banyak pengguna aktif yang akan berinteraksi dengan aplikasi Anda dalam jangka waktu tertentu. Misalnya, jika Anda mengharapkan aplikasi Anda masuk rata-rata satu juta pengguna aktif dalam periode delapan jam, maka Anda harus dapat mengautentikasi rata-rata 35 pengguna per detik.

Selain itu, jika Anda berasumsi bahwa sesi pengguna rata-rata adalah dua jam, dan Anda mengonfigurasi token untuk kedaluwarsa setelah satu jam, setiap pengguna harus menyegarkan token mereka sekali selama sesi mereka. Kuota rata-rata yang diperlukan untuk `UserAuthentication` kategori untuk mendukung beban ini adalah 70 RPS.

Jika Anda mengasumsikan peak-to-average rasio 3:1 dengan memperhitungkan varians frekuensi masuk pengguna selama periode delapan jam, maka Anda memerlukan kuota 200 RPS yang diinginkan `UserAuthentication`.

### Note

Jika Anda memanggil beberapa operasi untuk setiap tindakan pengguna, Anda harus meringkas tingkat panggilan operasi individual di tingkat kategori.

## Optimalkan tarif permintaan untuk batas kuota

Karena peningkatan batas tarif API menambah biaya pada AWS tagihan Anda, pertimbangkan penyesuaian model penggunaan Anda sebelum Anda meminta kenaikan kuota. Berikut ini adalah beberapa contoh arsitektur aplikasi yang mengoptimalkan tingkat permintaan.

Coba lagi upaya setelah periode tunggu mundur

Anda dapat menangkap error dengan setiap panggilan API, dan kemudian mencoba kembali upaya setelah periode back-off. Anda dapat menyesuaikan algoritma mundur sesuai dengan kebutuhan dan beban bisnis. SDK Amazon memiliki logika coba ulang bawaan. Untuk informasi selengkapnya, lihat [Alat untuk Dibangun AWS](#).

Gunakan database eksternal untuk atribut yang sering diperbarui

Jika aplikasi Anda memerlukan beberapa panggilan ke kolam pengguna untuk membaca atau menulis atribut kustom, gunakan penyimpanan eksternal. Anda dapat menggunakan basis data pilihan Anda untuk menyimpan atribut kustom atau menggunakan lapisan cache untuk memuat profil pengguna selama masuk. Anda dapat mereferensikan profil ini dari cache saat diperlukan, alih-alih memuat ulang profil pengguna dari kumpulan pengguna.

Validasi token web JSON (JWT) di sisi klien

Aplikasi harus memvalidasi token JWT sebelum mempercayainya. Anda dapat memverifikasi tanda tangan dan validitas token di sisi klien tanpa mengirim permintaan API ke kumpulan pengguna. Setelah token divalidasi, Anda dapat memercayai klaim dalam token dan menggunakan klaim alih-alih membuat lebih banyak Panggilan API `getUser`. Untuk informasi selengkapnya, lihat [Memverifikasi Token Web JSON](#).

Membatasi lalu lintas ke aplikasi web Anda dengan ruang tunggu

Jika Anda mengharapkan lalu lintas dari sejumlah besar pengguna yang masuk selama acara terikat waktu, seperti mengikuti ujian atau menghadiri acara langsung, Anda dapat mengoptimalkan lalu lintas permintaan dengan mekanisme self-throttling. Anda dapat, misalnya, menyiapkan ruang tunggu di mana pengguna dapat berdiri sampai sesi tersedia, memungkinkan Anda untuk memproses permintaan ketika Anda memiliki kapasitas yang tersedia. Lihat [solusi Ruang Tunggu AWS Virtual](#) untuk arsitektur referensi ruang tunggu.

Cache JWT

Gunakan kembali token akses hingga kedaluwarsa. Untuk contoh framework dengan token caching di API Gateway, lihat [Token cache](#). Alih-alih membuat permintaan API untuk menanyakan

informasi pengguna, cache ID token hingga kedaluwarsa, dan membaca atribut pengguna dari cache.

Untuk informasi selengkapnya tentang bekerja dengan tingkat permintaan API AWS, lihat [Mengelola dan memantau pembatasan API di beban kerja Anda](#). Untuk informasi tentang mengoptimalkan operasi Amazon Cognito yang menambahkan biaya ke tagihan AWS Anda, lihat [Mengelola biaya](#)

## Lacak penggunaan kuota

Amazon Cognito menghasilkan `CallCount` dan `ThrottleCount` metrik di Amazon CloudWatch untuk setiap kategori operasi API di tingkat akun. Anda dapat menggunakan `CallCount` untuk melacak jumlah total panggilan yang dilakukan pelanggan terkait dengan suatu kategori. Anda dapat menggunakan `ThrottleCount` untuk melacak jumlah total panggilan yang di-throttle yang terkait dengan suatu kategori. Anda dapat menggunakan metrik `CallCount` dan `ThrottleCount` dengan statistik `Sum` untuk menghitung jumlah total panggilan dalam suatu kategori. Untuk informasi selengkapnya, lihat [metrik CloudWatch penggunaan](#).

Saat memantau kuota layanan, pemanfaatan adalah persentase kuota layanan yang digunakan. Misalnya, jika nilai kuota 200 sumber daya, dan 150 sumber daya digunakan, pemanfaatannya adalah 75%. Penggunaan adalah jumlah sumber daya atau operasi yang digunakan untuk kuota layanan.

Melacak penggunaan melalui CloudWatch metrik

Anda dapat melacak dan mengumpulkan metrik pemanfaatan kumpulan pengguna Amazon Cognito. CloudWatch CloudWatchDasbor menampilkan metrik tentang setiap Layanan AWS yang Anda gunakan. Dengan CloudWatch, Anda dapat membuat alarm metrik untuk memberi tahu Anda atau mengubah sumber daya tertentu yang sedang Anda pantau. Untuk informasi selengkapnya tentang CloudWatch metrik, lihat [Melacak metrik CloudWatch penggunaan Anda](#).

Pelacakan pemanfaatan melalui metrik Service Quotas

Kumpulan pengguna Amazon Cognito terintegrasi dengan Service Quotas, antarmuka konsol untuk menampilkan dan mengelola penggunaan kuota layanan Anda. Di konsol Service Quotas, Anda dapat mencari nilai kuota tertentu, melihat informasi pemantauan, meminta peningkatan kuota, atau mengatur alarm. CloudWatch Setelah akun Anda aktif untuk sementara waktu, Anda dapat melihat grafik pemanfaatan sumber daya Anda.

Kolom nilai kuota tingkat akun Terapan di konsol Service Quotas untuk [kumpulan pengguna Amazon Cognito dan kumpulan identitas Amazon Cognito](#) menampilkan kuota Anda saat ini. Kolom Utilisasi menampilkan tingkat penggunaan kuota Anda saat ini. Kuota kumpulan pengguna Amazon Cognito requests-per-second (RPS) yang dapat disesuaikan menampilkan penggunaannya saat ini. Konsol Service Quotas juga dapat menavigasi Anda ke CloudWatch metrik untuk melihat lebih dekat pada metrik kuota yang dipilih. Untuk informasi selengkapnya tentang melihat kuota di konsol Service Quotas, lihat [Melihat Service Quotas](#).

## Lacak pengguna aktif bulanan (MAUs)

Jumlah pengguna aktif bulanan (MAUs) di kumpulan pengguna Anda menyumbangkan data penting untuk perencanaan Anda untuk peningkatan kuota tingkat permintaan. Anda dapat membandingkan tingkat permintaan API Anda dengan jumlah pengguna yang telah aktif dalam periode waktu tertentu. Dengan pengetahuan itu, Anda dapat menghitung bagaimana peningkatan pengguna aktif aplikasi Anda akan memengaruhi kuota Anda dalam model penggunaan Anda. Misalnya, bayangkan aplikasi gabungan Anda di US West (Oregon) menghasilkan 2 juta pengguna aktif dalam sebulan dan UserAuthentication kategori Anda menerima kesalahan pembatasan sesekali pada kuota default 120 permintaan per detik (RPS). Pada bulan sebelumnya, sebelum kampanye iklan Anda sukses, Anda memiliki 1 juta UI dan aplikasi Anda tidak pernah melebihi 80 RPS. Jika Anda mengantisipasi lonjakan serupa sebagai akibat dari tempat TV baru, Anda dapat membeli 40 RPS tambahan untuk mengakomodasi jutaan pengguna berikutnya dengan kuota yang disesuaikan 160 RPS.

Untuk meninjau MAUs Anda

Akses [AWS Billing konsol](#) dan tinjau tagihan terbaru. Berdasarkan biaya berdasarkan layanan, Anda dapat memfilter Cognito untuk melihat rincian UI Anda untuk periode penagihan tersebut.

## Meminta peningkatan kuota

Amazon Cognito memiliki kuota untuk jumlah maksimum operasi per detik yang dapat Anda lakukan di kumpulan pengguna dan kumpulan identitas di masing-masing. Wilayah AWS Anda dapat membeli peningkatan kuota tingkat permintaan API kumpulan pengguna Amazon Cognito yang dapat disesuaikan. Periksa kuota Anda saat ini dan beli peningkatan dari konsol Service Quotas atau dengan operasi Service Quotas API dan. `ListAWSDefaultServiceQuotas` `RequestServiceQuotaIncrease`

- Untuk membeli peningkatan kuota menggunakan konsol Service Quotas, [lihat Meminta peningkatan kuota API di Panduan Pengguna Service Quotas](#).



- AWS menargetkan penyelesaian permintaan peningkatan kuota dalam waktu 10 hari. Namun, beberapa pertimbangan dapat menyebabkan waktu pemrosesan permintaan melebihi 10 hari. Beberapa permintaan, misalnya, mungkin mengharuskan Amazon Cognito untuk menyediakan kapasitas perangkat keras tambahan, dan peningkatan volume permintaan musiman mungkin menyebabkan penundaan.
- Jika kuota tidak tersedia di Service Quotas, gunakan formulir peningkatan [batas Layanan](#).

#### Important

Hanya kuota yang dapat disesuaikan yang dapat ditingkatkan. Anda harus membeli peningkatan kapasitas kuota. Untuk harga kenaikan kuota, lihat harga [Amazon Cognito](#).

## Pengguna Amazon Cognito mengumpulkan kategori operasi API dan kuota tingkat permintaan

Karena Amazon Cognito memiliki kelas operasi API yang tumpang tindih dengan [model otorisasi yang berbeda](#), setiap operasi termasuk dalam kategori. Setiap kategori memiliki kuota gabungan sendiri untuk semua operasi API anggota, di semua kumpulan pengguna dalam satu Wilayah AWS di akun Anda. Anda hanya dapat meminta peningkatan kuota kategori yang dapat disesuaikan. Untuk informasi selengkapnya, lihat [Meminta peningkatan kuota](#). Penyesuaian kuota berlaku untuk kumpulan pengguna di akun Anda dalam satu Wilayah. Amazon Cognito membatasi operasi di beberapa kategori <sup>3</sup>hingga 5 permintaan per detik (RPS), per kumpulan pengguna. Kuota Default (RPS) juga berlaku untuk semua kumpulan pengguna dalam file. Akun AWS

#### Note

Kuota untuk setiap kategori diukur dalam Pengguna Aktif Bulanan (MAUs). Akun AWS dengan kurang dari dua juta UI dapat beroperasi dalam kuota default. Jika Anda memiliki kurang dari satu juta UI dan Amazon Cognito membatasi permintaan, pertimbangkan untuk mengoptimalkan aplikasi Anda. Untuk informasi selengkapnya, lihat [Optimalkan tarif permintaan untuk batas kuota](#).

Kuota operasi kategori diterapkan di semua pengguna di semua kumpulan pengguna dalam satu Wilayah AWS. Amazon Cognito juga mempertahankan kuota untuk jumlah permintaan yang dapat

dihasilkan aplikasi Anda terhadap satu pengguna. Anda harus membatasi permintaan API per pengguna seperti yang ditunjukkan pada tabel berikut.

Kuota tarif permintaan per pengguna Amazon Cognito

Operasi	Operasi per pengguna per detik
Baca profil pengguna Contoh: <code>GetUser</code> , <code>GetDevice</code>	10
Tulis profil pengguna Contoh: <code>UpdateUserAttributes</code> , <code>SetUserSettings</code>	10

Anda harus membatasi permintaan API per kategori seperti yang ditunjukkan pada tabel berikut.

Kuota tarif permintaan per kategori kumpulan pengguna Amazon Cognito

Kategori	Deskripsi	Kuota standar (RPS)	Dapat Disesuaikan
UserAuthentication	Operasi yang mengautentikasi (masuk) pengguna.	120	Ya
<ul style="list-style-type: none"> <li><a href="#">InitiateAuth</a></li> <li>Penyegaran token dengan <code>InitiateAuth</code> atau <a href="#">Titik akhir token</a></li> <li><a href="#">RespondToAuthChallenge</a><sup>1</sup></li> <li><a href="#">AdminInitiateAuth</a></li> <li><a href="#">AdminRespondToAuthChallenge</a><sup>1</sup></li> </ul>	Operasi ini tunduk pada <a href="#">Pengguna Amazon Cognito menggabungkan operasi API dengan penanganan tingkat permintaan khusus</a> .		

Kategori	Deskripsi	Kuota standar (RPS)	Dapat Disesuaikan
<ul style="list-style-type: none"> <li>Login UI yang di-host dan MFA <a href="#">dalam kode otorisasi</a> atau hibah implisit <sup>2</sup></li> </ul>			
<b>UserCreation</b> <ul style="list-style-type: none"> <li><a href="#">SignUp</a></li> <li><a href="#">ConfirmSignUp</a></li> <li><a href="#">AdminCreateUser</a></li> <li><a href="#">AdminConfirmSignUp</a></li> </ul>	Operasi yang membuat atau mengonfirmasi pengguna lokal Amazon Cognito. Ini adalah pengguna yang dibuat dan diverifikasi langsung oleh kumpulan pengguna Amazon Cognito Anda.	50	Ya
<b>UserFederation</b>  Operasi yang menggabungkan (mengautentikasi) pengguna dengan penyedia identitas pihak ketiga ke dalam kumpulan pengguna Amazon Cognito Anda.	Operasi yang mengirimkan respons IDP ke titik akhir federasi kumpulan pengguna. Operasi OIDC atau penyedia sosial yang menghasilkan token IDP, dan semua permintaan SAMP, berkontribusi pada kuota ini.	25	Ya

Kategori	Deskripsi	Kuota standar (RPS)	Dapat Disesuaikan
UserAccountRecovery <ul style="list-style-type: none"> <li>• <a href="#">ChangePassword</a></li> <li>• <a href="#">ConfirmForgotPassword</a></li> <li>• <a href="#">ForgotPassword</a></li> <li>• <a href="#">AdminResetUserPassword</a></li> <li>• <a href="#">AdminSetUserPassword</a></li> <li>• <a href="#">RespondToAuthChallenge<sup>1</sup></a></li> <li>• <a href="#">AdminRespondToAuthChallenge<sup>1</sup></a></li> <li>• <a href="#">Setel ulang kata sandi UI yang dihosting</a></li> </ul>	Operasi yang memulihkan akun pengguna, atau mengubah atau memperbarui kata sandi pengguna.	30	Tidak
UserRead <ul style="list-style-type: none"> <li>• <a href="#">AdminGetUser</a></li> <li>• <a href="#">GetUser</a></li> </ul>	Operasi yang mengambil pengguna dari kolam pengguna Anda.	120	Ya

Kategori	Deskripsi	Kuota standar (RPS)	Dapat Disesuaikan
UserUpdate <ul style="list-style-type: none"> <li>• <a href="#">AdminAddUserToGroup</a></li> <li>• <a href="#">AdminDeleteUserAttributes</a></li> <li>• <a href="#">AdminUpdateUserAttributes</a></li> <li>• <a href="#">AdminDeleteUser</a></li> <li>• <a href="#">AdminDisableUser</a></li> <li>• <a href="#">AdminEnableUser</a></li> <li>• <a href="#">AdminLinkProviderForUser</a></li> <li>• <a href="#">AdminDisableProviderForUser</a></li> <li>• <a href="#">VerifyUserAttribute</a></li> <li>• <a href="#">DeleteUser</a></li> <li>• <a href="#">DeleteUserAttributes</a></li> <li>• <a href="#">UpdateUserAttributes</a></li> <li>• <a href="#">AdminUserGlobalSignOut</a></li> <li>• <a href="#">GlobalSignOut</a></li> <li>• <a href="#">AdminRemoveUserFromGroup</a></li> </ul>	Operasi yang Anda gunakan untuk mengelola pengguna dan atribut pengguna.	25	Tidak
UserToken <ul style="list-style-type: none"> <li>• <a href="#">RevokeToken</a></li> </ul>	Operasi untuk manajemen token	120	Ya

Kategori	Deskripsi	Kuota standar (RPS)	Dapat Disesuaikan
UserResourceRead	Operasi yang mengambil informasi sumber daya pengguna dari Amazon Cognito, seperti perangkat yang diingat atau keanggotaan grup.	50	Ya
	<ul style="list-style-type: none"><li>• <a href="#">AdminGetDevice</a></li><li>• <a href="#">AdminListGroupsWithUser</a></li><li>• <a href="#">AdminListDevices</a></li><li>• <a href="#">GetDevice</a></li><li>• <a href="#">ListDevices</a></li><li>• <a href="#">GetUserAttributeVerificationCode</a></li><li>• <a href="#">ResendConfirmationCode</a></li><li>• <a href="#">AdminListUserAuthEvents</a></li></ul>		

Kategori	Deskripsi	Kuota standar (RPS)	Dapat Disesuaikan
UserResourceUpdate <ul style="list-style-type: none"> <li>• <a href="#">AdminForgetDevice</a></li> <li>• <a href="#">AdminUpdateAuthEventFeedback</a></li> <li>• <a href="#">AdminSetUserMFAReferensi</a></li> <li>• <a href="#">AdminSetUserSettings</a></li> <li>• <a href="#">AdminUpdateDeviceStatus</a></li> <li>• <a href="#">UpdateDeviceStatus</a></li> <li>• <a href="#">UpdateAuthEventFeedback</a></li> <li>• <a href="#">ConfirmDevice</a></li> <li>• <a href="#">SetUserMFAReferensi</a></li> <li>• <a href="#">SetUserSettings</a></li> <li>• <a href="#">VerifySoftwareToken</a></li> <li>• <a href="#">AssociateSoftwareToken</a></li> <li>• <a href="#">ForgetDevice</a></li> </ul>	Operasi yang memperbarui informasi sumber daya untuk pengguna, seperti perangkat yang diingat atau keanggotaan grup.	25	Tidak
UserList <ul style="list-style-type: none"> <li>• <a href="#">ListUsers</a></li> <li>• <a href="#">ListUsersInGroup</a></li> </ul>	Operasi yang mengembalikan daftar pengguna.	30	Tidak

Kategori	Deskripsi	Kuota standar (RPS)	Dapat Disesuaikan
UserPoolRead	Operasi yang membaca kolom pengguna Anda. <ul style="list-style-type: none"><li>• <a href="#">DescribeUserPool</a></li><li>• <a href="#">ListUserPools</a></li></ul>	15	Tidak
UserPoolUpdate	Operasi yang membuat, memperbarui, atau menghapus kumpulan pengguna Anda. <ul style="list-style-type: none"><li>• <a href="#">CreateUserPool</a></li><li>• <a href="#">UpdateUserPool</a></li><li>• <a href="#">DeleteUserPool</a></li></ul>	15	Tidak



Kategori	Deskripsi	Kuota standar (RPS)	Dapat Disesuaikan
UserPoolResourceRead	Operasi yang mengambil informasi tentang sumber daya, seperti grup atau server sumber daya, dari kumpulan pengguna. <sup>3</sup>	20	Tidak
	<ul style="list-style-type: none"> <li>• <a href="#">DescribeIdentityProvider</a></li> <li>• <a href="#">DescribeResourceServer</a></li> <li>• <a href="#">DescribeUserImportJob</a></li> <li>• <a href="#">DescribeUserPoolDomain</a></li> <li>• <a href="#">GetCSVHeader</a></li> <li>• <a href="#">GetGroup</a></li> <li>• <a href="#">GetSigningCertificate</a></li> <li>• <a href="#">GetIdentityProviderByIdentifier</a></li> <li>• <a href="#">GetUserPoolMfaConfig</a></li> <li>• <a href="#">ListGroup</a></li> <li>• <a href="#">ListIdentityProviders</a></li> <li>• <a href="#">ListResourceServers</a></li> <li>• <a href="#">ListTagsForResource</a></li> <li>• <a href="#">ListUserImportJobs</a></li> <li>• <a href="#">DescribeRiskConfiguration</a></li> <li>• <a href="#">GetuiCustomization</a></li> </ul>		

Kategori	Deskripsi	Kuota standar (RPS)	Dapat Disesuaikan
UserPoolResourceUpdate	Operasi yang memodifikasi sumber daya, seperti grup atau server sumber daya, dalam kumpulan pengguna.	15	Tidak
<ul style="list-style-type: none"> <li>• <a href="#">AddCustomAttribute</a></li> <li>• <a href="#">CreateGroup</a></li> <li>• <a href="#">CreateIdentityProvider</a></li> <li>• <a href="#">CreateResourceServer</a></li> <li>• <a href="#">CreateUserImportJob</a></li> <li>• <a href="#">CreateUserPoolDomain</a></li> <li>• <a href="#">DeleteGroup</a></li> <li>• <a href="#">DeleteIdentityProvider</a></li> <li>• <a href="#">DeleteResourceServer</a></li> <li>• <a href="#">DeleteUserPoolDomain</a></li> <li>• <a href="#">SetUserPoolMfaConfig</a></li> <li>• <a href="#">StartUserImportJob</a></li> <li>• <a href="#">StopUserImportJob</a></li> <li>• <a href="#">UpdateGroup</a></li> <li>• <a href="#">UpdateIdentityProvider</a></li> <li>• <a href="#">UpdateResourceServer</a></li> <li>• <a href="#">UpdateUserPoolDomain</a></li> </ul>	3		

Kategori	Deskripsi	Kuota standar (RPS)	Dapat Disesuaikan
<ul style="list-style-type: none"> <li>• <a href="#">SetRiskConfiguration</a></li> <li>• <a href="#">Setuicustomisasi</a></li> <li>• <a href="#">TagResource</a></li> <li>• <a href="#">UntagResource</a></li> </ul>			
UserPoolClientRead <ul style="list-style-type: none"> <li>• <a href="#">DescribeUserPoolClient</a></li> <li>• <a href="#">ListUserPoolClients</a></li> </ul>	Operasi yang mengambil informasi tentang klien kumpulan pengguna Anda. <sup>3</sup>	15	Tidak
UserPoolClientUpdate <ul style="list-style-type: none"> <li>• <a href="#">CreateUserPoolClient</a></li> <li>• <a href="#">DeleteUserPoolClient</a></li> <li>• <a href="#">UpdateUserPoolClient</a></li> </ul>	Operasi yang membuat, memperbaiki, dan menghapus klien kumpulan pengguna Anda. <sup>3</sup>	15	Tidak
ClientAuthentication client_credentials berikan permintaan tipe ke titik akhir token.	Operasi yang menghasilkan kredensi untuk digunakan dalam mengotorisasi permintaan machine-to-machine	150	Tidak

<sup>1</sup> A RespondToAuthChallenge atau AdminRespondToAuthChallenge tanggapan dengan ChallengeName NEW\_PASSWORD\_REQUIRED hitungan terhadap UserAccountRecovery kategori. Semua respons tantangan lainnya diperhitungkan dalam UserAuthentication kategori.

<sup>2</sup> Setiap operasi UI yang di-host selama login memberikan kontribusi satu permintaan ke kuota. Misalnya, pengguna yang masuk dan memberikan kode MFA menyumbangkan 2 permintaan. Penukaran token dalam hibah kode otorisasi tunduk pada alokasi kuota tambahan dengan tarif yang sama dengan kuota Anda dalam kategori. `UserAuthentication`

<sup>3</sup> Setiap operasi individu dalam kategori ini memiliki kendala yang mencegah operasi dipanggil pada tingkat yang lebih tinggi dari 5 RPS untuk kumpulan pengguna tunggal.

## Kumpulan identitas Amazon Cognito (identitas federasi) kuota tingkat permintaan operasi API

Operasi	Deskripsi	Kuota standar (RPS) <sup>1</sup>	Dapat Disesuaikan	Kuota meningkatkan kelayakan
<code>GetId</code>	Ambil ID identitas dari kumpulan identitas.	25	Ya	Hubungi tim akun Anda.
<code>GetOpenIdToken</code>	Ambil token OpenID dari kumpulan identitas dalam alur kerja klasik.	200	Ya	Hubungi tim akun Anda.
<code>GetCredentialsForIdentity</code>	Ambil AWS kredensial dari kumpulan identitas dalam alur kerja yang disempurnakan.	200	Ya	Hubungi tim akun Anda.
<code>GetOpenIdTokenForDeveloperIdentity</code>	Ambil token OpenID dari kumpulan identitas dalam	50	Ya	Hubungi tim akun Anda.

Operasi	Deskripsi	Kuota standar (RPS) <sup>1</sup>	Dapat Disesuaikan	Kuota meningkatkan kelayakan
	alur kerja pengembang.			
ListIdentities	Mengambil daftar ID identitas dalam kumpulan identitas.	5	Ya	Hubungi tim akun Anda.
DeleteIdentities	Hapus satu atau beberapa identitas terdaftar dari kumpulan identitas.	10	Ya	Hubungi tim akun Anda.
TagResource	Menerapkan tag ke kumpulan identitas.	5	Ya	Hubungi tim akun Anda.
UntagResource	Hapus tag dari kumpulan identitas.	5	Ya	Hubungi tim akun Anda.
ListTagsForResource	Menampilkan daftar tag yang diterapkan ke kumpulan identitas.	10	Ya	Hubungi tim akun Anda.

<sup>1</sup> Kuota default adalah kuota tingkat permintaan minimum untuk kumpulan identitas di salah satu Wilayah AWS di Anda. Akun AWS Kuota RPS Anda mungkin lebih tinggi di beberapa Wilayah.

## Kuota pada jumlah dan ukuran sumber daya

Kuota sumber daya adalah jumlah atau ukuran maksimum sumber daya, bidang input, durasi waktu, dan fitur lain-lain lainnya di Amazon Cognito.

Anda dapat meminta penyesuaian pada beberapa kuota sumber daya di konsol Service Quotas atau dari formulir peningkatan [batas Layanan](#). Untuk meminta kuota dari konsol Service Quotas, lihat [Meminta kenaikan kuota](#) di Panduan Pengguna Service Quotas. Jika kuota tidak tersedia di Service Quotas, gunakan formulir peningkatan [batas Layanan](#).

### Note

Kuota sumber daya di Akun AWS level tersebut, seperti kumpulan Pengguna per Wilayah, berlaku untuk sumber daya Amazon Cognito di masing-masing Wilayah AWS. Misalnya, Anda dapat memiliki 1.000 kumpulan pengguna di US East (Virginia N.) dan 1.000 lainnya di Eropa (Stockholm).

Tabel berikut menunjukkan kuota sumber daya default, dan apakah mereka dapat disesuaikan.

Kuota sumber daya kumpulan pengguna Amazon Cognito

Sumber Daya	Kuota	Dapat Disesuaikan	Kuota maksimum
Klien aplikasi per kumpulan pengguna	1.000	Ya	10.000
Kumpulan pengguna per Wilayah	1.000	Ya	10.000
Penyedia identitas per kolam pengguna	300	Ya	1.000
Server sumber daya per kolam pengguna	25	Ya	300
Pengguna per kumpulan pengguna	40.000.000	Ya	Hubungi tim akun Anda.

Sumber Daya	Kuota	Dapat Disesuaikan	Kuota maksimum
Total perubahan gabungan dalam pemicu Lambda generasi pra token <sup>1</sup>	5.000	Ya	Hubungi tim akun Anda.
Atribut khusus per kumpulan pengguna	50	Tidak	N/A
Karakter per atribut	2.048 byte	Tidak	N/A
Karakter dalam nama atribut kustom	20	Tidak	N/A
Karakter kata sandi minimum yang diperlukan dalam kebijakan kata sandi	6—99	Tidak	N/A
Pesan email dikirim setiap hari per Akun AWS <sup>2</sup>	50	Tidak	N/A
Karakter dalam subjek email	140	Tidak	N/A
Karakter dalam pesan email	20.000	Tidak	N/A
Karakter dalam pesan verifikasi SMS	140	Tidak	N/A
Karakter dalam kata sandi	256	Tidak	N/A
Karakter dalam nama penyedia identitas	32	Tidak	N/A

Sumber Daya	Kuota	Dapat Disesuaikan	Kuota maksimum
Pengidentifikasi per penyedia identitas	50	Tidak	N/A
Identitas yang ditautkan ke pengguna	5	Tidak	N/A
URL panggilan balik per klien aplikasi	100	Tidak	N/A
Logout URL per klien aplikasi	100	Tidak	N/A
Cakupan per server sumber daya	100	Tidak	N/A
Cakupan per klien aplikasi	50	Tidak	N/A
Domain kustom per akun	4	Tidak	N/A
Grup yang menjadi milik setiap pengguna	100	Tidak	N/A
Grup per kolam pengguna	10.000	Tidak	N/A

<sup>1</sup> Kuota ini mungkin ditemukan dalam token dari [Pemicu Lambda generasi pra token](#). Jumlah klaim yang ada dan yang ditambahkan ditambah cakupan dalam token akses dan identitas harus bertambah hingga angka yang lebih kecil dari atau sama dengan kuota ini. Klaim dan cakupan yang ditekan tidak berkontribusi pada kuota ini.

<sup>2</sup> Kuota ini hanya berlaku jika Anda menggunakan fitur email default untuk kumpulan pengguna Amazon Cognito. Untuk volume pengiriman email yang lebih tinggi, konfigurasi kumpulan pengguna Anda untuk menggunakan konfigurasi email Amazon SES Anda. Untuk informasi selengkapnya, lihat [Pengaturan email untuk kumpulan pengguna Amazon Cognito](#).



## Parameter validitas sesi kumpulan pengguna Amazon Cognito

Token	Kuota
Token ID	5 menit – 1 hari
Token refresh	1 jam – 3.650 hari
Token akses	5 menit – 1 hari
Cookie sesi UI yang dihosting	1 jam
Token sesi otentikasi	3 menit — 15 menit

Pengguna Amazon Cognito mengumpulkan kuota sumber daya keamanan kode (tidak dapat disesuaikan)

Sumber Daya	Kuota
Masa berlaku kode konfirmasi pendaftaran	24 jam
Periode validitas kode verifikasi atribut pengguna	24 jam
Periode validitas kode otentikasi multi-faktor (MFA)	3—15 menit
Lupa masa berlaku kode kata sandi	1 jam
Jumlah maksimum <code>ConfirmForgotPassword</code> dan <code>ForgotPassword</code> permintaan per pengguna per jam <sup>1</sup>	5—20
Jumlah maksimum <code>ResendConfirmationCode</code> permintaan per pengguna per jam	5
Jumlah maksimum <code>ConfirmSignUp</code> permintaan per pengguna per jam	15

Sumber Daya	Kuota
Jumlah maksimum ChangePassword permintaan per pengguna per jam	5
Jumlah maksimum GetUserAttributeVerificationCode permintaan per pengguna per jam	5
Jumlah maksimum VerifyUserAttribute permintaan per pengguna per jam	15

<sup>1</sup> Amazon Cognito mengevaluasi faktor risiko dalam permintaan untuk memperbarui kata sandi dan menetapkan kuota yang terkait dengan tingkat risiko yang dievaluasi. Untuk informasi selengkapnya, lihat [Perilaku lupa kata sandi](#).

Pengguna Amazon Cognito mengumpulkan kuota sumber daya pekerjaan impor pengguna

Sumber Daya	Kuota	Dapat Disesuaikan	Kuota maksimum
Tugas impor pengguna per kolam pengguna	1.000	Ya	Hubungi tim akun Anda.
Karakter maksimum per pengguna mengimpor baris CSV	16.000	Tidak	N/A
Ukuran file CSV maksimum	100 MB	Tidak	N/A
Jumlah maksimum pengguna per file CSV	500.000	Tidak	N/A

Kuota sumber daya kumpulan identitas Amazon Cognito (identitas federasi)

Sumber Daya	Kuota	Dapat Disesuaikan	Kuota maksimum
Kolam identitas per akun	1.000	Ya	N/A
Penyedia kumpulan pengguna Amazon Cognito per kumpulan identitas	50	Ya	1000
Panjang karakter dari nama kolam identitas	128 byte	Tidak	N/A
Panjang karakter dari nama penyedia login	2.048 byte	Tidak	N/A
Identitas per kumpulan identitas	Tidak terbatas.	Tidak	N/A
Penyedia identitas yang pemetaan perannya dapat ditentukan	10	Tidak	N/A
Hasil dari satu daftar atau panggilan pencarian	60	Tidak	N/A
Aturan kontrol akses berbasis peran (RBAC)	25	Tidak	N/A

### Kuota sumber daya Amazon Cognito Sync

Sumber Daya	Kuota	Dapat Disesuaikan	Kuota maksimum
Set data per identitas	20	Ya	Hubungi tim akun Anda.
Catatan per set data	1,024	Ya	Hubungi tim akun Anda.
Ukuran satu set data	1 MB	Ya	Hubungi tim akun Anda.
Karakter dalam nama dataset	128 byte	Tidak	N/A
Waktu tunggu untuk publikasi massal setelah permintaan berhasil	24 jam	Tidak	N/A

# API Amazon Cognito dan referensi titik akhir

Referensi berikut menjelaskan titik akhir layanan untuk setiap fitur Amazon Cognito. Kumpulan pengguna Amazon Cognito memiliki opsi berikut: [titik akhir kumpulan pengguna](#) dengan domain kumpulan pengguna, dan API kumpulan [pengguna](#). Untuk rincian kelas operasi API dengan API kumpulan pengguna kumpulan pengguna Amazon Cognito, lihat. [Menggunakan API kumpulan pengguna Amazon Cognito dan titik akhir kumpulan pengguna](#)

Untuk mengetahui daftar titik akhir layanan untuk API kumpulan pengguna Wilayah AWS, lihat [Titik akhir Layanan](#) di Referensi AWS Umum.

## Topik

- [Titik akhir federasi kumpulan pengguna dan referensi UI yang dihosting](#)
- [Referensi API kumpulan pengguna Amazon Cognito](#)
- [Referensi API kumpulan identitas Amazon Cognito \(identitas gabungan\)](#)
- [Referensi API sinkronisasi Amazon Cognito](#)

## Titik akhir federasi kumpulan pengguna dan referensi UI yang dihosting

Amazon Cognito mengaktifkan halaman web publik yang tercantum di sini saat Anda menetapkan domain ke kumpulan pengguna Anda. Domain Anda berfungsi sebagai titik akses pusat untuk semua klien aplikasi Anda. Mereka termasuk UI yang dihosting, tempat pengguna Anda dapat mendaftar dan masuk (the [Titik akhir masuk](#)), dan keluar (the [Titik akhir logout](#)). Untuk informasi lebih lanjut tentang sumber daya ini, lihat [Menyiapkan dan menggunakan UI yang dihosting Amazon Cognito dan titik akhir federasi](#).

Halaman-halaman ini juga menyertakan sumber daya web publik yang memungkinkan kumpulan pengguna Anda untuk berkomunikasi dengan SAMP pihak ketiga, OpenID Connect (OIDC) dan penyedia identitas OAuth 2.0 (). IdPs Untuk masuk pengguna dengan penyedia identitas federasi, pengguna Anda harus memulai permintaan ke UI yang dihosting interaktif [Titik akhir masuk](#) atau OIDC. [Otorisasi titik akhir](#) Titik akhir Otorisasi mengarahkan pengguna Anda ke UI yang dihosting atau halaman masuk IDP Anda.

Aplikasi Anda juga dapat masuk ke pengguna lokal dengan API [kumpulan pengguna Amazon Cognito](#). Pengguna lokal ada secara eksklusif di direktori kumpulan pengguna Anda tanpa federasi melalui iDP eksternal.

Selain UI yang dihosting dan titik akhir federasi, Amazon Cognito terintegrasi dengan SDK untuk Android, iOS JavaScript, dan banyak lagi. SDK menyediakan alat untuk melakukan operasi API kumpulan pengguna dengan titik akhir layanan Amazon Cognito API. Untuk informasi selengkapnya tentang titik akhir layanan, lihat titik akhir dan kuota [Identitas Amazon Cognito](#).

#### Warning

Jangan menyematkan sertifikat Transport Layer Security (TLS) entitas akhir atau menengah untuk domain Amazon Cognito. AWS mengelola semua sertifikat untuk semua titik akhir kumpulan pengguna dan domain awalan Anda. Otoritas sertifikat (CA) dalam rantai kepercayaan yang mendukung sertifikat Amazon Cognito secara dinamis berputar dan diperbarui. Saat Anda menyematkan aplikasi ke sertifikat perantara atau daun, aplikasi Anda dapat gagal tanpa pemberitahuan saat AWS memutar sertifikat.

Sebagai gantinya, sematkan aplikasi Anda ke semua [sertifikat root Amazon](#) yang tersedia. Untuk informasi selengkapnya, lihat praktik dan rekomendasi terbaik di [menyematkan Sertifikat](#) di Panduan AWS Certificate Manager Pengguna.

#### Topik

- [Referensi titik akhir UI yang di-host](#)
- [Referensi titik akhir federasi OAuth 2.0, OpenID Connect, dan SAMP 2.0](#)
- [Hibah OAuth 2.0](#)
- [Menggunakan PKCE dalam hibah kode otorisasi dengan kumpulan pengguna Amazon Cognito](#)
- [Respons kesalahan UI dan federasi yang dihosting](#)

## Referensi titik akhir UI yang di-host

Amazon Cognito mengaktifkan titik akhir UI yang dihosting di bagian ini saat Anda menambahkan domain ke kumpulan pengguna. Mereka adalah halaman web di mana pengguna Anda dapat menyelesaikan operasi otentikasi inti dari kumpulan pengguna. Mereka termasuk halaman untuk manajemen kata sandi, otentikasi multi-faktor (MFA), dan verifikasi atribut. Untuk informasi

selengkapnya tentang pengalaman pengguna di UI yang dihosting, lihat [Mendaftar dan masuk dengan UI yang dihosting](#).

Halaman web yang membentuk UI yang dihosting adalah aplikasi web front-end untuk sesi pengguna interaktif dengan pelanggan Anda. Aplikasi Anda harus memanggil UI yang dihosting di browser pengguna Anda. Amazon Cognito tidak mendukung akses terprogram ke halaman web di bagian ini. Titik akhir federasi [Referensi titik akhir federasi OAuth 2.0, OpenID Connect, dan SAMP 2.0](#) yang menampilkan respons JSON dapat ditanyakan langsung dalam kode aplikasi Anda. [Otorisasi titik akhir](#) Pengalihan ke UI yang dihosting atau ke halaman masuk IDP dan juga harus dibuka di browser pengguna.

Topik dalam panduan ini menjelaskan titik akhir UI yang sering digunakan secara rinci. Amazon Cognito membuat halaman web yang mengikuti tersedia saat Anda menetapkan domain ke kumpulan pengguna Anda.

Titik akhir UI yang dihosting

URL Titik Akhir	Deskripsi	Bagaimana itu diakses
<a href="https://Domain_kumpulan_pengguna_Anda/login">https://Domain_kumpulan_pengguna_Anda/login</a>	Masuk ke kumpulan pengguna pengguna lokal dan federasi.	Redirect dari endpoint seperti <a href="#">Otorisasi titik akhir</a> , /logout, dan. /confirmf orgotPassword Lihat <a href="#">Titik akhir masuk</a> .
<a href="https://Domain_kumpulan_pengguna_Anda/logout">https://Domain_kumpulan_pengguna_Anda/logout</a>	Keluar pengguna kumpulan pengguna.	Tautan langsung. Lihat <a href="#">Titik akhir logout</a> .
<a href="https://Domain_kumpulan_pengguna_Anda/ConfirmUser">https://Domain_kumpulan_pengguna_Anda/ConfirmUser</a>	Mengonfirmasi pengguna yang telah memilih tautan email untuk memverifikasi akun pengguna mereka.	Tautan yang dipilih pengguna dalam pesan email.
<a href="https://Domain_kumpulan_pengguna_Anda/pendaftaran">https://Domain_kumpulan_pengguna_Anda/pendaftaran</a>	Mendaftar pengguna baru. /loginHalaman mengarahkan pengguna Anda /signup saat mereka memilih Daftar.	Tautan langsung dengan parameter yang sama seperti /oauth2/authorize .

URL Titik Akhir	Deskripsi	Bagaimana itu diakses
<a href="https://Domain_kumpulan_pengguna_Anda/konfirmasi">https://Domain_kumpulan_pengguna_Anda/konfirmasi</a>	Setelah kumpulan pengguna Anda mengirimkan kode konfirmasi ke pengguna yang mendaftar, minta kode tersebut kepada pengguna Anda.	Hanya pengalihan dari. / signup
<a href="https://Domain_kumpulan_pengguna_Anda/ForgotPassword">https://Domain_kumpulan_pengguna_Anda/ForgotPassword</a>	Meminta pengguna Anda untuk nama pengguna mereka dan mengirimkan kode pengaturan ulang kata sandi. /loginHalaman mengarahkan pengguna Anda /forgotPassword ketika mereka memilih Lupa kata sandi Anda? .	<ol style="list-style-type: none"> <li>1. Dari tautan Lupa kata sandi di/login.</li> <li>2. Tautan langsung dengan parameter yang sama seperti/oauth2/authorize .</li> </ol>
<a href="https://Domain_kumpulan_pengguna_Anda/confirmForgotPassword">https://Domain_kumpulan_pengguna_Anda/confirmForgotPassword</a>	Meminta pengguna Anda untuk kode pengaturan ulang kata sandi dan kata sandi baru. /forgotPassword Halaman mengarahkan pengguna Anda /confirmforgotPassword saat mereka memilih Reset kata sandi Anda.	Hanya pengalihan dari. / forgotPassword
<a href="https://Domain_kumpulan_pengguna_Anda/resetcode">https://Domain_kumpulan_pengguna_Anda/resetcode</a>	Mengirim kode konfirmasi baru ke pengguna yang telah mendaftar di kumpulan pengguna Anda.	Hanya pengalihan dari Kirim tautan kode baru di. / confirm

## Topik

- [Titik akhir masuk](#)



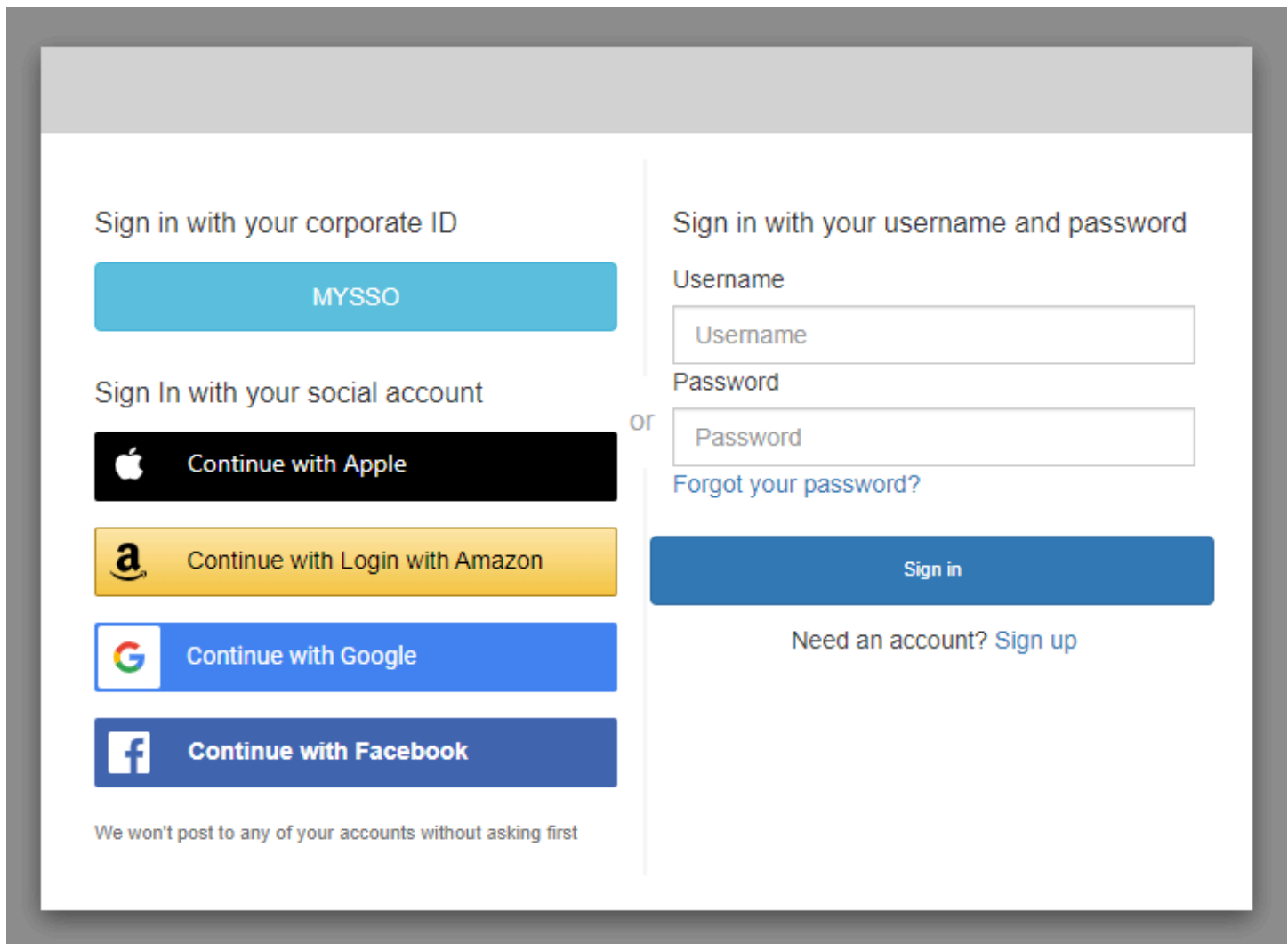
- [Titik akhir logout](#)

## Titik akhir masuk

Endpoint login adalah server otentikasi dan tujuan pengalihan dari [Otorisasi titik akhir](#). Ini adalah titik masuk ke UI yang dihosting saat Anda tidak menentukan penyedia identitas. Saat Anda membuat pengalihan ke titik akhir login, itu memuat halaman login dan menyajikan opsi otentikasi yang dikonfigurasi untuk klien kepada pengguna.

### Note

Endpoint login adalah komponen dari UI yang dihosting. Di aplikasi Anda, panggil federasi dan halaman UI yang dihosting yang mengarahkan ulang ke titik akhir login. Akses langsung oleh pengguna ke titik akhir login bukanlah praktik terbaik.



GET /login

/loginTitik akhir hanya mendukung HTTPS GET permintaan awal pengguna Anda. Aplikasi Anda memanggil halaman di browser seperti Chrome atau Firefox. Ketika Anda mengarahkan ke /login dari [Otorisasi titik akhir](#), itu melewati semua parameter yang Anda berikan dalam permintaan awal Anda. Endpoint login mendukung semua parameter permintaan dari titik akhir otorisasi. Anda juga dapat mengakses titik akhir login secara langsung. Sebagai praktik terbaik, mulailah semua sesi pengguna Anda di. /oauth2/authorize

Contoh - meminta pengguna untuk masuk

Contoh ini menampilkan layar login.

```
GET https://mydomain.auth.us-east-1.amazoncognito.com/login?
    response_type=code&
```

```
client_id=ad398u21ijw3s9w3939&
redirect_uri=https://YOUR_APP/redirect_uri&
state=STATE&
scope=openid+profile+aws.cognito.signin.user.admin
```

## Contoh - respon

Server autentikasi mengalihkan ke aplikasi Anda dengan kode otorisasi dan status. Server harus mengembalikan kode dan status dalam parameter string kueri dan bukan di fragmen.

```
HTTP/1.1 302 Found
      Location: https://YOUR_APP/redirect_uri?
code=AUTHORIZATION_CODE&state=STATE
```

## Permintaan masuk yang diprakarsai pengguna

Setelah pengguna Anda memuat `/login` titik akhir, mereka dapat memasukkan nama pengguna dan kata sandi dan memilih Masuk. Ketika mereka melakukan ini, mereka menghasilkan HTTPS POST permintaan dengan parameter permintaan header yang sama dengan GET permintaan, dan badan permintaan dengan nama pengguna, kata sandi, dan sidik jari perangkat mereka.

## Titik akhir logout

Titik `/logout` akhir adalah titik akhir pengalihan. Ini mengeluarkan pengguna dan mengalihkan ke URL keluar resmi untuk klien aplikasi Anda, atau ke titik akhir. `/login` Parameter yang tersedia dalam permintaan GET ke `/logout` titik akhir disesuaikan dengan kasus penggunaan UI yang dihosting Amazon Cognito.

Untuk mengarahkan pengguna Anda ke UI yang dihosting untuk masuk lagi, tambahkan `redirect_uri` parameter ke permintaan Anda. `logout` Permintaan dengan `redirect_uri` parameter juga harus menyertakan parameter untuk permintaan Anda berikutnya ke [Titik akhir masuk](#), seperti `client_id`, `response_type`, dan `scope`.

Endpoint logout adalah aplikasi web front-end untuk sesi pengguna interaktif dengan pelanggan Anda. Aplikasi Anda harus memanggil ini dan titik akhir UI yang dihosting lainnya di browser pengguna Anda.

Untuk mengarahkan pengguna ke halaman yang Anda pilih, tambahkan URL keluar yang Diizinkan ke klien aplikasi Anda. Dalam permintaan pengguna Anda ke `logout` titik akhir, tambahkan

`logout_uri` dan `client_id` parameter. Jika nilai `logout_uri` adalah salah satu URL keluar yang Diizinkan untuk klien aplikasi Anda, Amazon Cognito mengarahkan pengguna ke URL tersebut.

Dengan single logout (SLO) untuk SAMP 2.0, Amazon IdPs Cognito pertama-tama mengalihkan pengguna Anda ke titik akhir SLO yang Anda tentukan dalam konfigurasi iDP Anda. Setelah iDP mengalihkan pengguna Anda kembali ke, Amazon `saml2/logout` Cognito merespons dengan satu pengalihan lagi ke atau dari permintaan Anda. `redirect_uri` `logout_uri` Untuk informasi selengkapnya, lihat [Alur keluar SAMP](#).

Titik akhir logout tidak mengeluarkan pengguna dari OIDC atau penyedia identitas sosial (). IdPs Untuk mengeluarkan pengguna dari sesi mereka dengan iDP eksternal, arahkan mereka ke halaman keluar untuk penyedia tersebut.

GET `/logout`

Titik akhir `/logout` hanya mendukung HTTPS GET. Klien kumpulan pengguna biasanya membuat permintaan ini melalui browser sistem. Browser biasanya Custom Chrome Tab di Android atau Safari View Control di iOS.

Permintaan parameter

`client_id`

ID klien aplikasi untuk aplikasi Anda. Untuk mendapatkan ID klien aplikasi, Anda harus mendaftarkan aplikasi di kumpulan pengguna. Untuk informasi selengkapnya, lihat [Klien aplikasi kumpulan pengguna](#).

Diperlukan.

`logout_uri`

Arahkan pengguna Anda ke halaman keluar khusus dengan parameter `logout_uri`. Tetapkan nilainya ke URL keluar klien aplikasi tempat Anda ingin mengarahkan pengguna setelah mereka keluar. Gunakan `logout_uri` hanya dengan parameter `client_id`. Untuk informasi selengkapnya, lihat [Klien aplikasi kumpulan pengguna](#).

Anda juga dapat menggunakan parameter `logout_uri` untuk mengarahkan pengguna ke halaman login untuk klien aplikasi lain. Setel halaman login untuk klien aplikasi lain sebagai URL callback yang Diizinkan di klien aplikasi Anda. Dalam permintaan Anda ke `/logout` titik akhir, tetapkan nilai parameter `logout_uri` ke halaman login yang disandikan URL.

Amazon Cognito memerlukan `logout_uri` atau parameter `redirect_uri` dalam permintaan Anda ke titik akhir. `/logout` Parameter `logout_uri` mengarahkan pengguna Anda ke situs web lain. Jika parameter `logout_uri` dan `redirect_uri` disertakan dalam permintaan Anda ke titik akhir, `/logout` Amazon Cognito akan menggunakan parameter `logout_uri` secara eksklusif, mengesampingkan parameter `redirect_uri`.

### `redirect_uri`

Arahkan pengguna Anda ke halaman login Anda untuk mengautentikasi dengan parameter `redirect_uri`. Tetapkan nilainya ke URL callback yang diizinkan klien aplikasi tempat Anda ingin mengarahkan pengguna setelah mereka masuk lagi. Tambahkan parameter `client_id`, `scope`, `state`, dan `response_type` yang ingin Anda lewatkan ke titik akhir Anda. `/login`

Amazon Cognito memerlukan `logout_uri` atau parameter `redirect_uri` dalam permintaan Anda ke titik akhir. `/logout` Untuk mengarahkan pengguna ke `/login` titik akhir Anda untuk mengautentikasi ulang dan meneruskan token ke aplikasi Anda, tambahkan parameter `redirect_uri`. Jika parameter `logout_uri` dan `redirect_uri` disertakan dalam permintaan Anda ke titik akhir, `/logout` Amazon Cognito akan mengganti parameter `redirect_uri` dan memproses parameter `logout_uri` secara eksklusif.

### `response_type`

Respons OAuth 2.0 yang ingin Anda terima dari Amazon Cognito setelah pengguna masuk. `codedan token` merupakan nilai yang valid untuk parameter `response_type`.

Diperlukan jika Anda menggunakan parameter `redirect_uri`.

### `status`

Saat aplikasi Anda menambahkan parameter `status` ke permintaan, Amazon Cognito mengembalikan nilainya ke aplikasi Anda saat `/oauth2/logout` titik akhir mengalihkan pengguna Anda.

Tambahkan nilai ini ke permintaan Anda untuk menjaga terhadap serangan [CSRF](#).

Anda tidak dapat mengatur nilai `state` parameter ke string JSON yang dikodekan URL. Untuk meneruskan string yang cocok dengan format ini dalam `state` parameter, encode string ke `base64`, lalu decode dalam aplikasi Anda.

Sangat disarankan jika Anda menggunakan parameter `redirect_uri`.

## cakupan

Cakupan OAuth 2.0 yang ingin Anda minta dari Amazon Cognito setelah Anda mengeluarkannya dengan parameter `redirect_uri`. Amazon Cognito mengalihkan pengguna Anda ke `/login` titik akhir dengan parameter cakupan dalam permintaan Anda ke titik akhir. `/logout`

Opsional jika Anda menggunakan parameter `redirect_uri`. Jika Anda tidak menyertakan parameter cakupan, Amazon Cognito mengalihkan pengguna Anda ke `/login` titik akhir dengan parameter cakupan. Saat Amazon Cognito mengalihkan pengguna Anda dan mengisi secara otomatis `scope`, parameter tersebut menyertakan semua cakupan resmi untuk klien aplikasi Anda.

## Permintaan contoh

Contoh - log out dan mengarahkan pengguna ke klien

Kecuali untuk `logout_uri` dan `client_id`, semua parameter kueri yang mungkin untuk titik akhir ini diteruskan ke. [Otorisasi titik akhir](#) Amazon Cognito mengalihkan sesi pengguna ke URL dalam nilai `logout_uri`, mengabaikan semua parameter permintaan lainnya, saat permintaan menyertakan `logout_uri` `client_id` URL ini harus berupa URL keluar resmi untuk klien aplikasi.

Berikut ini adalah contoh permintaan untuk keluar dan redirect ke. `https://www.example.com/welcome`

```
GET https://mydomain.auth.us-east-1.amazoncognito.com/logout?  
client_id=1example23456789&  
logout_uri=https%3A%2F%2Fwww.example.com%2Fwelcome
```

Contoh - keluar dan minta pengguna untuk masuk sebagai pengguna lain

Ketika permintaan dihilangkan `logout_uri` tetapi sebaliknya memberikan parameter yang membentuk permintaan yang dibentuk dengan baik ke titik akhir otorisasi, Amazon Cognito mengarahkan pengguna ke login UI yang dihosting. Titik akhir logout menambahkan parameter dalam permintaan asli Anda ke tujuan pengalihan. Parameter `redirect_uri` dalam permintaan ke titik akhir logout bukanlah URL keluar, tetapi URL login yang ingin Anda lewati ke titik akhir otorisasi.

Berikut ini adalah contoh permintaan yang menandatangani pengguna keluar, mengalihkan ke halaman login, dan memberikan kode otorisasi setelah login. `https://www.example.com`

```
GET https://mydomain.auth.us-east-1.amazoncognito.com/logout?  
response_type=code&
```

```
client_id=1example23456789&
redirect_uri=https%3A%2F%2Fwww.example.com&
state=example-state-value&
nonce=example-nonce-value&
scope=openid+profile+aws.cognito.signin.user.admin
```

## Referensi titik akhir federasi OAuth 2.0, OpenID Connect, dan SAMP 2.0

Amazon Cognito mengaktifkan titik akhir di bagian ini saat Anda menambahkan domain ke kumpulan pengguna. Titik akhir federasi tidak interaktif pengguna. Mereka menjalankan peran layanan untuk aplikasi Anda untuk berkomunikasi dengan penyedia identitas OAuth 2.0, OIDC, dan SAMP 2.0 pihak ketiga (). IdPs

Topik dalam panduan ini menjelaskan beberapa titik akhir OAuth 2.0 dan OIDC yang sering digunakan. Amazon Cognito membuat titik akhir berikut saat Anda menetapkan domain ke kumpulan pengguna.

Titik akhir federasi kumpulan pengguna

URL Titik Akhir	Deskripsi	Bagaimana itu diakses
<code>https://<i>Domain kumpulan pengguna Anda</i>/oauth2/otorisasi</code>	Mengalihkan pengguna ke UI yang dihosting atau masuk dengan IDP mereka.	Dipanggil di browser pelanggan untuk memulai otentikasi pengguna. Lihat <a href="#">Otorisasi titik akhir</a> .
<code>https://<i>Domain kumpulan pengguna Anda</i>/oauth2/token</code>	Mengembalikan token berdasarkan kode otorisasi atau permintaan kredensial klien.	Diminta oleh aplikasi untuk mengambil token. Lihat <a href="#">Titik akhir token</a> .
<code>https://<i>Domain kumpulan pengguna Anda</i>/OAuth2/userInfo</code>	Mengembalikan atribut pengguna berdasarkan cakupan OAuth 2.0 dan identitas pengguna dalam token akses.	Diminta oleh aplikasi untuk mengambil profil pengguna. Lihat <a href="#">Titik akhir UserInfo</a> .
<code>https://<i>Domain kumpulan pengguna Anda</i>/oauth2/callback</code>	Mencabut token penyegaran dan token akses terkait.	Diminta oleh aplikasi untuk mencabut token. Lihat <a href="#">Cabut titik akhir</a> .

URL Titik Akhir	Deskripsi	Bagaimana itu diakses
<code>https://cognito-idp. <i>Region</i> .amazonaws.com/ ID kumpulan pengguna Anda /.well-known/openid-configuration</code>	Direktori arsitektur OIDC dari kumpulan pengguna Anda.	Diminta oleh aplikasi untuk menemukan metadata penerbit kumpulan pengguna.
<code>https://cognito-idp. <i>Wilayah</i> .amazonaws.com/ ID kumpulan pengguna Anda /.well-known/jwks.json</code>	Kunci publik yang dapat Anda gunakan untuk memvalidasi token Amazon Cognito.	Diminta oleh aplikasi untuk memverifikasi JWT.
<code>https://<i>Domain kumpulan pengguna Anda</i> /oauth2/idpresponse</code>	Penyedia identitas sosial harus mengarahkan pengguna Anda ke titik akhir ini dengan kode otorisasi. Amazon Cognito menukarkan kode untuk token saat mengautentikasi pengguna federasi Anda.	Dialihkan dari OIDC iDP sign-in sebagai URL callback klien iDP.
<code>https://<i>Domain kumpulan pengguna Anda</i> /saml2/idpresponse</code>	URL Assertion Consumer Response (ACS) untuk integrasi dengan penyedia identitas SAMP 2.0.	Dialihkan dari SAMP 2.0 iDP sebagai URL ACS, atau titik originasi untuk login yang diprakarsai IDP. <sup>1</sup>
<code>https://<i>Domain kumpulan pengguna Anda</i> /saml2/logout</code>	URL <a href="#">Single Logout</a> (SLO) untuk integrasi dengan penyedia identitas SAMP 2.0.	Dialihkan dari SAMP 2.0 iDP sebagai URL logout tunggal (SLO). Hanya menerima pengikatan POST.

<sup>1</sup> Untuk informasi selengkapnya tentang login SAMP yang diprakarsai IDP, lihat. [Menggunakan login SAMP yang diprakarsai IDP](#)



[Untuk informasi selengkapnya tentang standar OpenID Connect dan OAuth, lihat OpenID Connect 1.0 dan OAuth 2.0.](#)

## Topik

- [Otorisasi titik akhir](#)
- [Titik akhir token](#)
- [Titik akhir UserInfo](#)
- [Cabut titik akhir](#)
- [titik akhir saml2/idpresponse](#)

## Otorisasi titik akhir

Titik `/oauth2/authorize` akhir adalah titik akhir pengalihan yang mendukung dua tujuan pengalihan. Jika Anda menyertakan `idp_identifier` parameter `identity_provider` atau di URL, parameter tersebut secara diam-diam mengarahkan pengguna Anda ke halaman masuk untuk penyedia identitas tersebut (iDP). Jika tidak, itu mengalihkan ke [Titik akhir masuk](#) dengan parameter URL yang sama yang Anda sertakan dalam permintaan Anda.

Titik akhir otorisasi mengalihkan ke UI yang dihosting atau ke halaman masuk IDP. Tujuan sesi pengguna di titik akhir ini adalah halaman web yang harus berinteraksi langsung dengan pengguna Anda di browser mereka.

Untuk menggunakan titik akhir otorisasi, panggil browser pengguna Anda di `/oauth2/authorize` dengan parameter yang menyediakan kumpulan pengguna Anda dengan informasi tentang detail kumpulan pengguna berikut.

- Klien aplikasi yang ingin Anda masuki.
- URL callback yang ingin Anda dapatkan.
- Cakupan OAuth 2.0 yang ingin Anda minta di token akses pengguna Anda.
- Secara opsional, idP pihak ketiga yang ingin Anda gunakan untuk masuk.

Anda juga dapat menyediakan `state` dan `nonce` parameter yang digunakan Amazon Cognito untuk memvalidasi klaim yang masuk.

## DAPATKAN `/oauth2/authorize`

Titik akhir `/oauth2/authorize` hanya mendukung HTTPS GET. Aplikasi Anda biasanya memulai permintaan ini di browser pengguna Anda. Anda hanya dapat membuat permintaan ke `/oauth2/authorize` titik akhir melalui HTTPS.

[Anda dapat mempelajari lebih lanjut tentang definisi titik akhir otorisasi dalam standar OpenID Connect \(OIDC\) di Authorization Endpoint.](#)

Permintaan parameter

### **response\_type**

(Wajib) Jenis respons. Harus code atau token.

Permintaan yang berhasil dengan `response_type` code pengembalian hibah kode otorisasi. Pemberian kode otorisasi adalah code parameter yang ditambahkan Amazon Cognito ke URL pengalihan Anda. Aplikasi Anda dapat bertukar kode dengan token akses, ID, dan penyegaran. [Titik akhir token](#) Sebagai praktik keamanan terbaik, dan untuk menerima token penyegaran bagi pengguna Anda, gunakan hibah kode otorisasi di aplikasi Anda.

Permintaan yang berhasil dengan `response_type` token pengembalian hibah implisit. Pemberian implisit adalah ID dan token akses yang ditambahkan Amazon Cognito ke URL pengalihan Anda. Hibah implisit kurang aman karena mengekspos token dan informasi identifikasi potensial kepada pengguna. Anda dapat menonaktifkan dukungan untuk hibah implisit dalam konfigurasi klien aplikasi Anda.

### **client\_id**

(Wajib) ID klien aplikasi.

Nilai `client_id` harus berupa ID klien aplikasi di kumpulan pengguna tempat Anda membuat permintaan. Klien aplikasi Anda harus mendukung proses masuk oleh pengguna lokal Amazon Cognito atau setidaknya satu IDP pihak ketiga.

### **redirect\_uri**

(Wajib) URL tempat server otentikasi mengalihkan browser setelah Amazon Cognito memberi otorisasi kepada pengguna.

Pengenal sumber daya seragam pengalihan (URI) harus memiliki atribut berikut:

- Ini harus URI mutlak.

- Anda harus telah melakukan pra-registrasi URI dengan klien.
- Itu tidak dapat menyertakan komponen fragmen.

Lihat [OAuth 2.0 - Titik Akhir Pengalihan](#).

Amazon Cognito mengharuskan URI pengalihan Anda menggunakan HTTPS, kecuali `http://localhost`, yang dapat Anda tetapkan sebagai URL panggilan balik untuk tujuan pengujian.

Amazon Cognito juga mendukung URL panggilan balik aplikasi seperti `myapp://example`

## **state**

(Opsional, disarankan) Saat aplikasi Anda menambahkan parameter status ke permintaan, Amazon Cognito mengembalikan nilainya ke aplikasi Anda saat `/oauth2/authorize` titik akhir mengarahkan pengguna Anda.

Tambahkan nilai ini ke permintaan Anda untuk menjaga terhadap serangan [CSRF](#).

Anda tidak dapat mengatur nilai `state` parameter ke string JSON yang dikodekan URL. Untuk meneruskan string yang cocok dengan format ini dalam `state` parameter, encode string ke base64, lalu dekodekannya di aplikasi Anda.

## **identity\_provider**

(Opsional) Tambahkan parameter ini untuk melewati UI yang dihosting dan mengarahkan pengguna Anda ke halaman login penyedia. Nilai parameter `identity_provider` adalah nama penyedia identitas (idP) seperti yang muncul di kumpulan pengguna Anda.

- Untuk penyedia sosial, Anda dapat menggunakan nilai `identity_providerFacebook`, `Google`, `LoginWithAmazon` dan `SignInWithApple`
- Untuk kumpulan pengguna Amazon Cognito, gunakan nilainya. `COGNITO`
- Untuk penyedia identitas SAMP 2.0 dan OpenID Connect (OIDC) (IdPs), gunakan nama yang Anda tetapkan ke idP di kumpulan pengguna Anda.

## **idp\_identifier**

(Opsional) Tambahkan parameter ini untuk mengarahkan ke penyedia dengan nama alternatif untuk nama `identity_provider`. Anda dapat memasukkan pengenalan untuk SAMP 2.0 dan OIDC IdPs dari tab Pengalaman masuk di konsol Amazon Cognito.

## **scope**

(Opsional) Dapat berupa kombinasi dari cakupan yang dicadangkan sistem atau cakupan khusus yang terkait dengan klien. Lingkup harus dipisahkan oleh spasi. Cakupan sistem cadangan adalah

`openid`, `email`, `phone`, `profile`, dan `aws.cognito.signin.user.admin`. Lingkup yang digunakan harus dikaitkan dengan klien, atau akan diabaikan pada saat waktu aktif.

Jika klien tidak meminta cakupan apa pun, server autentikasi menggunakan semua cakupan yang terkait dengan klien.

Token ID hanya dikembalikan jika cakupan `openid` diminta. Token akses hanya dapat digunakan terhadap kolam pengguna Amazon Cognito jika cakupan `aws.cognito.signin.user.admin` diminta. Cakupan `phone`, `email`, dan `profile` hanya dapat diminta jika cakupan `openid` juga diminta. Lingkup ini mendikte klaim yang masuk ke dalam token ID.

### **code\_challenge\_method**

(Opsional) Protokol hashing yang Anda gunakan untuk menghasilkan tantangan. [PKCE RFC](#) mendefinisikan dua metode, S256 dan polos; namun, server autentikasi Amazon Cognito hanya mendukung S256.

### **code\_challenge**

(Opsional) Tantangan yang Anda hasilkan dari `code_verifier`.

Diperlukan hanya ketika Anda menentukan `code_challenge_method` parameter.

### **nonce**

(Opsional) Nilai acak yang dapat Anda tambahkan ke permintaan. Nilai nonce yang Anda berikan disertakan dalam token ID yang dikeluarkan Amazon Cognito. Untuk mencegah serangan replay, aplikasi Anda dapat memeriksa nonce klaim dalam token ID dan membandingkannya dengan yang Anda buat. Untuk informasi selengkapnya tentang nonce klaim, lihat [validasi token ID dalam standar](#) OpenID Connect.

Contoh permintaan dengan tanggapan positif

Contoh berikut menggambarkan format permintaan HTTP ke titik `/oauth2/authorize` akhir.

Pemberian kode otorisasi

Ini adalah contoh permintaan untuk hibah kode otorisasi.

Contoh - DAPATKAN permintaan

Permintaan berikut memulai sesi untuk mengambil kode otorisasi yang diteruskan pengguna ke aplikasi Anda di tujuan. `redirect_uri` Sesi ini meminta cakupan atribut pengguna dan akses ke operasi API layanan mandiri Amazon Cognito.

```
GET https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/authorize?
response_type=code&
client_id=1example23456789&
redirect_uri=https://www.example.com&
state=abcdefg&
scope=openid+profile+aws.cognito.signin.user.admin
```

### Contoh - respon

Server autentikasi Amazon Cognito mengalihkan kembali ke aplikasi Anda dengan kode otorisasi dan status. Kode otorisasi berlaku selama lima menit.

```
HTTP/1.1 302 Found
Location: https://www.example.com?code=a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111&state=abcdefg
```

### Pemberian kode otorisasi dengan PKCE

Ini adalah contoh permintaan untuk hibah kode otorisasi dengan [PKCE](#).

### Contoh - DAPATKAN permintaan

Permintaan berikut menambahkan `code_challenge` parameter ke permintaan sebelumnya. Untuk menyelesaikan pertukaran kode untuk token, Anda harus menyertakan `code_verifier` parameter dalam permintaan Anda ke `/oauth2/token` titik akhir.

```
GET https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/authorize?
response_type=code&
client_id=1example23456789&
redirect_uri=https://www.example.com&
state=abcdefg&
scope=aws.cognito.signin.user.admin&
code_challenge_method=S256&
code_challenge=a1b2c3d4...
```

### Contoh - respon

Server otentikasi mengalihkan kembali ke aplikasi Anda dengan kode otorisasi dan status. Kode dan status harus dikembalikan dalam parameter string kueri dan bukan di fragmen:

```
HTTP/1.1 302 Found
```

```
Location: https://www.example.com?code=a1b2c3d4-5678-90ab-cdef-EXAMPLE11111&state=abcdefg
```

## Pemberian token tanpa cakupan **openid**

Ini adalah contoh permintaan yang menghasilkan hibah implisit dan mengembalikan JWT langsung ke sesi pengguna.

### Contoh - DAPATKAN permintaan

Permintaan berikut adalah hibah implisit dari server otorisasi Anda. Token akses dari Amazon Cognito mengotorisasi operasi API swalayan.

```
GET https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/authorize?
response_type=token&
client_id=1example23456789&
redirect_uri=https://www.example.com&
state=abcdefg&
scope=aws.cognito.signin.user.admin
```

### Contoh - respon

Server otorisasi Amazon Cognito mengarahkan kembali ke aplikasi Anda dengan token akses. Karena **openid** cakupan tidak diminta, Amazon Cognito tidak mengembalikan token ID. Selain itu, Amazon Cognito tidak mengembalikan token penyegaran dalam aliran ini. Amazon Cognito mengembalikan token akses dan status dalam fragmen dan bukan dalam string kueri:

```
HTTP/1.1 302 Found
Location: https://YOUR_APP/
redirect_uri#access_token=ACCESS_TOKEN&token_type=bearer&expires_in=3600&state=STATE
```

## Pemberian token dengan cakupan **openid**

Ini adalah contoh permintaan yang menghasilkan hibah implisit dan mengembalikan JWT langsung ke sesi pengguna.

### Contoh - DAPATKAN permintaan

Permintaan berikut adalah hibah implisit dari server otorisasi Anda. Token akses dari Amazon Cognito mengotorisasi akses ke atribut pengguna dan operasi API layanan mandiri.

```
GET
https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/authorize?
response_type=token&
client_id=1example23456789&
redirect_uri=https://www.example.com&
state=abcdefg&
scope=aws.cognito.signin.user.admin+openid+profile
```

## Contoh - respon

Server otorisasi mengalihkan kembali ke aplikasi Anda dengan token akses dan token ID (karena openid cakupan telah disertakan):

```
HTTP/1.1 302 Found
Location: https://
www.example.com#id_token=eyJra67890EXAMPLE&access_token=eyJra12345EXAMPLE&token_type=bearer&exp
```

## Contoh tanggapan negatif

Amazon Cognito mungkin menolak permintaan Anda. Permintaan negatif datang dengan kode kesalahan HTTP dan deskripsi yang dapat Anda gunakan untuk memperbaiki parameter permintaan Anda. Berikut ini adalah contoh tanggapan negatif.

- `redirect_uri` Jika `client_id` dan valid, tetapi parameter permintaan tidak diformat dengan benar, server otorisasi mengalihkan kesalahan ke klien `redirect_uri` dan menambahkan pesan kesalahan dalam parameter URL. Berikut ini adalah contoh pemformatan yang salah.
  - Permintaan tidak menyertakan `response_type` parameter.
  - Permintaan otorisasi menyediakan `code_challenge` parameter, tetapi bukan `code_challenge_method` parameter.
  - Nilai `code_challenge_method` parameter tidak S256.

Berikut ini adalah respons terhadap permintaan contoh dengan format yang salah.

```
HTTP 1.1 302 Found Location: https://client_redirect_uri?error=invalid_request
```

- Jika klien meminta code atau token masuk `response_type`, tetapi tidak memiliki izin untuk permintaan ini, server otorisasi Amazon Cognito kembali `unauthorized_client` ke klien `redirect_uri`, sebagai berikut:

```
HTTP 1.1 302 Found Location: https://client_redirect_uri?error=unauthorized_client
```

- Jika klien meminta cakupan yang tidak diketahui, cacat, atau tidak valid, server otorisasi Amazon Cognito `invalid_scope` kembali ke `redirect_uri` klien, sebagai berikut:

```
HTTP 1.1 302 Found Location: https://client_redirect_uri?error=invalid_scope
```

- Jika ada kesalahan tak terduga di server, server otentikasi kembali `server_error` ke klien. `redirect_uri` Karena kesalahan HTTP 500 tidak dikirim ke klien, kesalahan tidak ditampilkan di browser pengguna. Server otorisasi mengembalikan kesalahan berikut.

```
HTTP 1.1 302 Found Location: https://client_redirect_uri?error=server_error
```

- Saat Amazon Cognito mengautentikasi melalui federasi ke pihak ketiga, Amazon IdPs Cognito mungkin mengalami masalah koneksi, seperti berikut ini:
  - Jika batas waktu koneksi terjadi saat meminta token dari IDP, server otentikasi mengalihkan kesalahan ke klien sebagai berikut: `redirect_uri`

```
HTTP 1.1 302 Found Location: https://client_redirect_uri?
error=invalid_request&error_description=Timeout+occurred+in+calling+IdP+token
+endpoint
```

- Jika batas waktu koneksi terjadi saat memanggil `jwtks_uri` titik akhir untuk validasi token ID, server otentikasi mengalihkan dengan kesalahan ke klien sebagai berikut: `redirect_uri`

```
HTTP 1.1 302 Found Location: https://client_redirect_uri?
error=invalid_request&error_description=error_description=Timeout+in+calling+jwks
+uri
```

- Saat mengautentikasi dengan mengfederasi ke pihak ketiga IdPs, penyedia dapat mengembalikan respons kesalahan. Ini bisa disebabkan oleh kesalahan konfigurasi atau alasan lain, seperti berikut ini:
  - Jika respons kesalahan diterima dari penyedia lain, server autentikasi mengalihkan kesalahan ke `redirect_uri` klien sebagai berikut:

```
HTTP 1.1 302 Found Location: https://client_redirect_uri?
error=invalid_request&error_description=[IdP name]+Error+-+[status code]+error
getting token
```



- Jika respons kesalahan diterima dari Google, server otentikasi mengalihkan kesalahan ke klien `redirect_uri` sebagai berikut:

```
HTTP 1.1 302 Found Location: https://client_redirect_uri?
error=invalid_request&error_description=Google+Error+--[status code]+[Google-
provided error code]
```

- Saat Amazon Cognito menemukan pengecualian komunikasi saat terhubung ke iDP eksternal, server autentikasi mengalihkan dengan kesalahan ke klien dengan salah satu pesan berikut: `redirect_uri`

```
HTTP 1.1 302 Found Location: https://client_redirect_uri?
error=invalid_request&error_description=Connection+reset
```

```
HTTP 1.1 302 Found Location: https://client_redirect_uri?
error=invalid_request&error_description=Read+timed+out
```

## Titik akhir token

[Titik akhir token OAuth 2.0 /oauth2/token mengeluarkan token](#) web JSON (JWT).

Server otorisasi OAuth 2.0 kumpulan pengguna Anda mengeluarkan token web JSON (JWT) dari titik akhir token ke jenis sesi berikut:

1. Pengguna yang telah menyelesaikan permintaan untuk pemberian kode otorisasi. Penukaran kode yang berhasil mengembalikan ID, akses, dan token penyegaran.
2. Sesi Machine-to-machine (M2M) yang telah menyelesaikan hibah kredensial klien. Otorisasi yang berhasil dengan rahasia klien mengembalikan token akses.
3. Pengguna yang sebelumnya telah masuk dan menerima token penyegaran. Segarkan otentikasi token mengembalikan ID baru dan token akses.

### Note

Pengguna yang masuk dengan pemberian kode otorisasi di UI yang dihosting atau melalui federasi selalu dapat menyegarkan token mereka dari titik akhir token. Pengguna yang masuk dengan operasi API `InitiateAuth` dan `AdminInitiateAuth` dapat menyegarkan token mereka dengan titik akhir token saat [perangkat yang diingat](#) tidak aktif di kumpulan pengguna Anda. Jika perangkat yang diingat aktif, segarkan token

dengan permintaan AuthFlow of REFRESH\_TOKEN\_AUTH in InitiateAuth atau AdminInitiateAuth API.

Titik akhir token menjadi tersedia untuk umum saat Anda menambahkan domain ke kumpulan pengguna Anda. Ia menerima permintaan HTTP POST. Untuk keamanan aplikasi, gunakan PKCE dengan peristiwa login kode otorisasi Anda. PKCE memverifikasi bahwa pengguna yang melewati kode otorisasi adalah pengguna yang sama yang diautentikasi. Untuk informasi lebih lanjut tentang PKCE, lihat [IETF RFC 7636](#).

Anda dapat mempelajari lebih lanjut tentang klien aplikasi kumpulan pengguna dan jenis hibah, rahasia klien, cakupan resmi, dan ID klien mereka di [Klien aplikasi kumpulan pengguna](#). Anda dapat mempelajari lebih lanjut tentang otorisasi M2M, hibah kredensial klien, dan otorisasi dengan cakupan token akses di [Cakupan, M2M, dan otorisasi API dengan server sumber daya](#)

Untuk mengambil informasi tentang pengguna dari token aksesnya, teruskan ke permintaan Anda [Titik akhir UserInfo](#) atau ke [GetUserAPI](#).

POST /oauth2/token

Titik akhir /oauth2/token hanya mendukung HTTPS POST. Aplikasi Anda membuat permintaan ke titik akhir ini secara langsung, bukan melalui browser pengguna.

Titik akhir token mendukung `client_secret_basic` dan `client_secret_post` otentikasi. Untuk informasi selengkapnya tentang spesifikasi OpenID Connect, lihat Otentikasi [Klien](#). Untuk informasi selengkapnya tentang titik akhir token dari spesifikasi OpenID Connect, [lihat](#) Titik Akhir Token.

Minta parameter di header

## Authorization

Jika klien dikeluarkan rahasia, klien dapat melewati `client_id` dan `client_secret` di header otorisasi sebagai otorisasi `client_secret_basic` HTTP. Anda juga dapat memasukkan `client_id` dan `client_secret` dalam badan permintaan sebagai `client_secret_post` otorisasi.

String header otorisasi adalah `BasicBase64Encode(client_id:client_secret)`. Contoh berikut adalah header otorisasi untuk klien aplikasi `djc98u3jiedmi283eu928` dengan rahasia klien `abcdef01234567890`, menggunakan versi string yang diencode Base64:  
`djc98u3jiedmi283eu928:abcdef01234567890`

```
Authorization: Basic ZGpj0Th1M2ppZWRtaTI4M2V10TI40mFiY2RlZjAxMjM0NTY3ODkw
```

## Content-Type

Tetapkan nilai parameter ini ke 'application/x-www-form-urlencoded'.

Minta parameter dalam tubuh

## grant\_type

(Wajib) Jenis hibah OIDC yang ingin Anda minta.

Harus `authorization_code` atau `refresh_token` atau `client_credentials`. Anda dapat meminta token akses untuk cakupan kustom dari titik akhir token dalam kondisi berikut:

- Anda mengaktifkan cakupan yang diminta dalam konfigurasi klien aplikasi Anda.
- Anda mengonfigurasi klien aplikasi Anda dengan rahasia klien.
- Anda mengaktifkan pemberian kredensi klien di klien aplikasi Anda.

## client\_id

(Opsional) ID klien aplikasi di kumpulan pengguna Anda. Tentukan klien aplikasi yang sama yang mengautentikasi pengguna Anda.

Anda harus memberikan parameter ini jika klien bersifat publik dan tidak memiliki rahasia, atau dengan `client_secret` `client_secret_post` otorisasi.

## client\_secret

(Opsional) Rahasia klien untuk klien aplikasi yang mengautentikasi pengguna Anda. Diperlukan jika klien aplikasi Anda memiliki rahasia klien dan Anda tidak mengirim Authorization header.

## scope

(Opsional) Dapat berupa kombinasi cakupan kustom apa pun yang terkait dengan klien aplikasi. Cakupan apa pun yang Anda minta harus diaktifkan untuk klien aplikasi. Jika tidak, Amazon Cognito akan mengabaikannya. Jika klien tidak meminta cakupan apa pun, server autentikasi akan menetapkan semua cakupan kustom yang Anda otorisasi dalam konfigurasi klien aplikasi.

Hanya digunakan jika `grant_type` adalah `client_credentials`.

## **redirect\_uri**

(Opsional) Harus sama dengan `redirect_uri` yang digunakan untuk `authorization_code` masuk/`oauth2/authorize`.

Anda harus memberikan parameter ini jika `grant_type` ada `authorization_code`.

## **refresh\_token**

(Opsional) Untuk menghasilkan token akses dan ID baru untuk sesi pengguna, tetapkan nilai `refresh_token` parameter dalam `/oauth2/token` permintaan Anda ke token penyegaran yang dikeluarkan sebelumnya dari klien aplikasi yang sama.

## **code**

(Opsional) Kode otorisasi dari hibah kode otorisasi. Anda harus memberikan parameter ini jika permintaan otorisasi Anda termasuk `grant_type`. `authorization_code`

## **code\_verifier**

(Opsional) Nilai arbitrer yang Anda gunakan untuk menghitung `code_challenge` dalam permintaan hibah kode otorisasi dengan PKCE.

Contoh permintaan dengan tanggapan positif

Menukar kode otorisasi untuk token

Contoh - permintaan POST

```
POST https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/token&
      Content-Type='application/x-www-form-urlencoded'&

Authorization=Basic ZGpj0Th1M2ppZWRtaTI4M2V1OTI4OmFiY2RlZjAxMjM0NTY3ODkw

      grant_type=authorization_code&
      client_id=1example23456789&
      code=AUTHORIZATION_CODE&
      redirect_uri=com.myclientapp://myclient/redirect
```

Contoh - respon

```
HTTP/1.1 200 OK

      Content-Type: application/json
```

```
{
  "access_token": "eyJra1example",
  "id_token": "eyJra2example",
  "refresh_token": "eyJj3example",
  "token_type": "Bearer",
  "expires_in": 3600
}
```

### Note

Titik akhir token mengembalikan refresh\_token hanya ketika grant\_type adalah authorization\_code.

Bertukar kredensial klien untuk token akses: rahasia klien di header otorisasi

Contoh - permintaan POST

```
POST https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/token >
      Content-Type='application/x-www-form-urlencoded'&

Authorization=Basic ZGpj0Th1M2ppZWRtaTI4M2V1OTI4OmFiY2RlZjAxMjM0NTY3ODkw

      grant_type=client_credentials&
      client_id=1example23456789&

scope=resourceServerIdentifier1/scope1 resourceServerIdentifier2/scope2
```

Contoh - respon

```
HTTP/1.1 200 OK

      Content-Type: application/json

      {
        "access_token": "eyJra1example",
        "token_type": "Bearer",
        "expires_in": 3600
      }
```

## Bertukar kredensial klien untuk token akses: rahasia klien di badan permintaan

### Contoh - permintaan POST

```
POST /oauth2/token HTTP/1.1
Content-Type: application/x-www-form-urlencoded
X-Amz-Target: AWSCognitoIdentityProviderService.Client_credentials_request
User-Agent: USER_AGENT
Accept: /
Accept-Encoding: gzip, deflate, br
Content-Length: 177
Referer: http://auth.example.com/oauth2/token
Host: auth.example.com
Connection: keep-alive

grant_type=client_credentials&client_id=1example23456789&scope=my_resource_server_identifier%2F
```

### Contoh - respon

```
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
Date: Tue, 05 Dec 2023 16:11:11 GMT
x-amz-cognito-request-id: 829f4fe2-a1ee-476e-b834-5cd85c03373b

{
  "access_token": "eyJra12345EXAMPLE",
  "expires_in": 3600,
  "token_type": "Bearer"
}
```

## Menukar hibah kode otorisasi dengan PKCE untuk token

### Contoh - permintaan POST

```
POST https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/token
      Content-Type='application/x-www-form-urlencoded'&

Authorization=Basic ZGpj0Th1M2ppZWRtaTI4M2V10TI40mFiY2RLZjAxMjM0NTY3ODkw

      grant_type=authorization_code&
      client_id=1example23456789&
      code=AUTHORIZATION_CODE&
      code_verifier=CODE_VERIFIER&
```

```
redirect_uri=com.myclientapp://myclient/redirect
```

## Contoh - respon

```
HTTP/1.1 200 OK
```

```
Content-Type: application/json
```

```
{  
  "access_token": "eyJra1example",  
  "id_token": "eyJra2example",  
  "refresh_token": "eyJj3example",  
  "token_type": "Bearer",  
  "expires_in": 3600  
}
```

### Note

Titik akhir token mengembalikan refresh\_token hanya ketika grant\_type adalah authorization\_code.

## Menukar token penyegaran dengan token

### Contoh - permintaan POST

```
POST https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/token >
```

```
Content-Type='application/x-www-form-urlencoded'&
```

```
Authorization=Basic ZGpj0Th1M2ppZWRtaTI4M2V10TI40mFiY2RlZjAxMjM0NTY3ODkw
```

```
grant_type=refresh_token&  
client_id=1example23456789&  
refresh_token=eyJj3example
```

## Contoh - respon

```
HTTP/1.1 200 OK
```

```
Content-Type: application/json
```

```
{
  "access_token": "eyJra1example",
  "id_token": "eyJra2example",
  "token_type": "Bearer",
  "expires_in": 3600
}
```

### Note

Titik akhir token mengembalikan `refresh_token` hanya ketika `grant_type` adalah `authorization_code`.

Contoh tanggapan negatif

Contoh - respon kesalahan

```
HTTP/1.1 400 Bad Request
```

```
Content-Type: application/json;charset=UTF-8
```

```
{
  "error": "invalid_request|invalid_client|invalid_grant|
unauthorized_client|unsupported_grant_type"
}
```

### **invalid\_request**

Permintaan tidak memiliki parameter yang diperlukan, menyertakan nilai parameter yang tidak didukung (selain `unsupported_grant_type`), atau formatnya salah. Misalnya, `grant_type` adalah `refresh_token` tetapi `refresh_token` tidak disertakan.

### **invalid\_client**

Autentikasi klien gagal. Misalnya, ketika klien menyertakan `client_id` dan `client_secret` di header otorisasi, tetapi tidak ada klien dengan `client_id` dan `client_secret` itu.

### **invalid\_grant**

Token refresh telah dicabut.

Kode otorisasi telah digunakan atau tidak ada.



Klien aplikasi tidak memiliki akses baca ke semua [atribut](#) dalam lingkup yang diminta. Misalnya, aplikasi Anda meminta email cakupan dan klien aplikasi Anda dapat membaca email atribut, tetapi tidak `email_verified`.

### **unauthorized\_client**

Klien tidak diperbolehkan untuk alur pemberian kode atau untuk untuk menyegarkan token.

### **unsupported\_grant\_type**

Dikembalikan jika `grant_type` ada sesuatu selain `authorization_code` atau `refresh_token` atau `client_credentials`.

## Titik akhir UserInfo

Titik `userInfo` akhir adalah titik akhir UserInfo OpenID Connect ([OIDC](#)). Ini merespons dengan atribut pengguna ketika penyedia layanan menyajikan token akses yang Anda [Titik akhir token](#) keluarkan. Cakupan dalam token akses pengguna Anda menentukan atribut pengguna yang ditampilkan oleh titik akhir UserInfo dalam responsnya. Ruang `openid` lingkup harus menjadi salah satu klaim token akses.

Amazon Cognito mengeluarkan token akses sebagai respons terhadap permintaan API kumpulan pengguna seperti [InitiateAuth](#). Karena tidak mengandung cakupan apa pun, `userInfo` titik akhir tidak menerima token akses ini. Sebagai gantinya, Anda harus menunjukkan token akses dari titik akhir token Anda.

Penyedia identitas pihak ketiga (iDP) OAuth 2.0 Anda juga menghosting titik akhir `userInfo`. Saat pengguna Anda mengautentikasi dengan IDP tersebut, Amazon Cognito secara diam-diam menukar kode otorisasi dengan endpoint iDP. token Kumpulan pengguna Anda meneruskan token akses IDP untuk mengotorisasi pengambilan informasi pengguna dari titik akhir iDP. `userInfo`

GET /oauth2/userInfo

Aplikasi Anda membuat permintaan ke titik akhir ini secara langsung dan bukan melalui browser.

Untuk informasi selengkapnya, lihat [UserInfoEndpoint](#) dalam spesifikasi OpenID Connect (OIDC).

### Topik

- [Minta parameter di header](#)
- [Contoh - permintaan](#)
- [Contoh — respon positif](#)

- [Contoh tanggapan negatif](#)

Minta parameter di header

**Authorization: Bearer <access\_token>**

Lulus token akses di bidang header otorisasi.

Wajib.

Contoh - permintaan

```
GET /oauth2/userInfo HTTP/1.1
Content-Type: application/x-amz-json-1.1
Authorization: Bearer eyJra12345EXAMPLE
User-Agent: [User agent]
Accept: */*
Host: auth.example.com
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
```

Contoh — respon positif

```
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
Content-Length: [Integer]
Date: [Timestamp]
x-amz-cognito-request-id: [UUID]
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
Cache-Control: no-cache, no-store, max-age=0, must-revalidate
Pragma: no-cache
Expires: 0
Strict-Transport-Security: max-age=31536000 ; includeSubDomains
X-Frame-Options: DENY
Server: Server
Connection: keep-alive
{
  "sub": "[UUID]",
  "email_verified": "true",
  "custom:mycustom1": "CustomValue",
  "phone_number_verified": "true",
```

```
"phone_number": "+12065551212",  
"email": "bob@example.com",  
"username": "bob"  
}
```

Untuk daftar klaim OIDC, lihat Klaim [Standar](#). Saat ini, Amazon Cognito mengembalikan nilai untuk `email_verified` dan `phone_number_verified` sebagai string.

Contoh tanggapan negatif

Contoh - permintaan buruk

```
HTTP/1.1 400 Bad Request  
WWW-Authenticate: error="invalid_request",  
error_description="Bad OAuth2 request at UserInfo Endpoint"
```

### **invalid\_request**

Permintaan tidak memiliki parameter yang diperlukan, termasuk nilai parameter yang tidak didukung, atau jika tidak, salah bentuk.

Contoh — token buruk

```
HTTP/1.1 401 Unauthorized  
WWW-Authenticate: error="invalid_token",  
error_description="Access token is expired, disabled, or deleted, or the user has  
globally signed out."
```

### **invalid\_token**

Token akses kedaluwarsa, dicabut, cacat, atau tidak valid.

## Cabut titik akhir

`oauth2/revoke` Titik akhir/mencabut token akses pengguna yang awalnya dikeluarkan Amazon Cognito dengan token penyegaran yang Anda berikan. Titik akhir ini juga mencabut semua token akses dan identitas berikutnya dari token penyegaran yang sama. Setelah titik akhir mencabut token, Anda tidak dapat menggunakan token akses yang dicabut untuk mengakses API yang diautentikasi oleh token Amazon Cognito.

## POST /oauth2/mencabut

Titik akhir /oauth2/revoke hanya mendukung HTTPS POST. Klien kolam pengguna membuat permintaan ke titik akhir ini secara langsung dan tidak melalui peramban sistem.

Minta parameter di header

### Authorization

Jika klien aplikasi Anda memiliki rahasia klien, aplikasi harus meneruskannya `client_id` dan `client_secret` di header otorisasi melalui otorisasi HTTP Dasar. Rahasiannya adalah [Dasar Base64Encode\(client\\_id:client\\_secret\)](#).

### Content-Type

Harus selalu 'application/x-www-form-urlencoded'.

Minta parameter dalam tubuh

### token

(Wajib) Token penyegaran yang ingin dicabut klien. Permintaan tersebut juga mencabut semua token akses yang dikeluarkan Amazon Cognito dengan token penyegaran ini.

Wajib.

### client\_id

(Opsional) ID klien aplikasi untuk token yang ingin Anda cabut.

Diperlukan jika klien bersifat publik dan tidak memiliki rahasia.

Contoh permintaan pencabutan

Contoh 1: Mencabut token untuk klien aplikasi tanpa rahasia klien

```
POST /oauth2/revoke HTTP/1.1
Host: https://mydomain.auth.us-east-1.amazoncognito.com
Accept: application/json
Content-Type: application/x-www-form-urlencoded
token=2YotnFZFEjr1zCsicMWpAA&
```

```
client_id=djc98u3jjiedmi283eu928
```

## Contoh 2: Mencabut token untuk klien aplikasi dengan rahasia klien

```
POST /oauth2/revoke HTTP/1.1
Host: https://mydomain.auth.us-east-1.amazoncognito.com
Accept: application/json
Content-Type: application/x-www-form-urlencoded
Authorization: Basic czZCaGRSa3F0MzpnWDFmQmF0M2JW
token=2YotnFZFEjr1zCsicMWpAA
```

## Respons kesalahan pencabutan

Respons yang berhasil berisi tubuh kosong. Respons kesalahan adalah objek JSON dengan `error` bidang dan, dalam beberapa kasus, `error_description` bidang.

## Kesalahan titik akhir

- Jika token tidak ada dalam permintaan atau jika fitur dinonaktifkan untuk klien aplikasi, Anda menerima HTTP 400 dan kesalahan `invalid_request`.
- Jika token yang dikirim Amazon Cognito dalam permintaan pencabutan bukan token penyegaran, Anda menerima HTTP 400 dan kesalahan. `unsupported_token_type`
- Jika kredensial klien tidak valid, Anda menerima HTTP 401 dan kesalahan. `invalid_client`
- Jika token telah dicabut atau jika klien mengirimkan token yang tidak valid, Anda menerima HTTP 200 OK.

## titik akhir saml2/idpresponse

Mereka `/saml2/idpresponse` menerima pernyataan SAMP. Dalam proses masuk service-provider-initiated (dimulai SP), penyedia identitas SAMP 2.0 (iDP) mengarahkan pengguna Anda ke titik akhir ini dengan respons SAMP mereka. Dalam login yang dimulai SP, aplikasi Anda tidak berinteraksi dengan titik akhir ini. Konfigurasi IDP Anda dengan jalur ke URL layanan konsumen pernyataan (ACS) Anda `saml2/idpresponse`. Untuk informasi selengkapnya tentang inisiasi sesi, lihat [Inisiasi sesi SAMP di kumpulan pengguna Amazon Cognito](#).

Dalam login yang dimulai IDP, pengguna Anda dapat masuk dengan IDP Anda melalui proses Anda sendiri dan mengirimkan pernyataan SAMP di badan permintaan melalui HTTPS. HTTP POST Isi

POST permintaan Anda harus berupa SAMLResponse parameter dan RelayState parameter. Untuk informasi selengkapnya, lihat [Menggunakan login SAMP yang diprakarsai IDP](#).

## POSTING `/saml2/idpresponse`

Untuk menggunakan `/saml2/idpresponse` titik akhir dalam login yang dimulai IDP, buat permintaan POST dengan parameter yang menyediakan kumpulan pengguna Anda dengan informasi tentang sesi pengguna Anda.

- Klien aplikasi yang ingin mereka masuki.
- URL callback yang mereka inginkan berakhir di.
- Cakupan OAuth 2.0 yang ingin mereka minta di token akses pengguna Anda.
- IdP yang memulai permintaan masuk.

Parameter badan permintaan yang diprakarsai IDP

### SamlResponse

Pernyataan SAMP yang diencode Base64 dari IDP yang terkait dengan klien aplikasi yang valid dan konfigurasi IDP di kumpulan pengguna Anda.

### RelayState

RelayStateParameter berisi parameter permintaan yang seharusnya Anda lewatkan ke `oauth2/authorize` titik akhir. Untuk informasi rinci tentang parameter ini, lihat [Otorisasi titik akhir](#).

`response_type`

Jenis hibah OAuth 2.0.

`client_id`

ID klien aplikasi.

`redirect_uri`

URL tempat server otentikasi mengalihkan browser setelah Amazon Cognito memberi otorisasi kepada pengguna.

`identity_provider`

Nama penyedia identitas tempat Anda ingin mengarahkan pengguna Anda.

## idp\_identifier

Pengidentifikasi penyedia identitas tempat Anda ingin mengarahkan pengguna Anda. cakupan

Cakupan OAuth 2.0 yang Anda ingin pengguna Anda minta dari server otorisasi.

Contoh permintaan dengan tanggapan positif

Contoh - permintaan POST

Permintaan berikut adalah pemberian kode otorisasi untuk pengguna dari MySAMLIDP IDP di klien aplikasi. `1example23456789` Pengguna mengalihkan ke `https://www.example.com` dengan kode otorisasi mereka, yang dapat ditukar dengan token yang menyertakan token akses dengan cakupan OAuth 2.0, dan openid email phone

```
POST /saml2/idpresponse HTTP/1.1
User-Agent: USER_AGENT
Accept: */*
Host: example.auth.us-east-1.amazoncognito.com
Content-Type: application/x-www-form-urlencoded
```

```
SAMLResponse=[Base64-encoded SAML assertion]&RelayState=identity_provider
%3DMySAMLIDP%26client_id%3D1example23456789%26redirect_uri%3Dhttps%3A%2F
%2Fwww.example.com%26response_type%3Dcode%26scope%3Demail%2Bopenid%2Bphone
```

Contoh - respon

Berikut ini adalah respons terhadap permintaan sebelumnya.

```
HTTP/1.1 302 Found
Date: Wed, 06 Dec 2023 00:15:29 GMT
Content-Length: 0
x-amz-cognito-request-id: 8aba6eb5-fb54-4bc6-9368-c3878434f0fb
Location: https://www.example.com?code=[Authorization code]
```

## Hibah OAuth 2.0

[Server otorisasi OAuth 2.0 kumpulan pengguna Amazon Cognito mengeluarkan token sebagai tanggapan atas tiga jenis hibah otorisasi OAuth 2.0.](#) Anda dapat menyetel jenis hibah yang didukung untuk setiap klien aplikasi di kumpulan pengguna Anda. Anda tidak dapat mengaktifkan hibah

kredensial klien di klien aplikasi yang sama seperti hibah kode implisit atau otorisasi. Permintaan untuk hibah kode implisit dan otorisasi dimulai dari Anda [Otorisasi titik akhir](#) dan permintaan untuk hibah kredensial klien dimulai dari Anda. [Titik akhir token](#)

## Pemberian kode otorisasi

Menanggapi permintaan otentikasi Anda yang berhasil, server otorisasi menambahkan kode otorisasi dalam code parameter ke URL callback Anda. Anda kemudian harus menukar kode untuk ID, akses, dan refresh token dengan [Titik akhir token](#). Untuk meminta pemberian kode otorisasi, setel `response_type` ke code dalam permintaan Anda. Untuk contoh permintaan, lihat [Pemberian kode otorisasi](#).

Pemberian kode otorisasi adalah bentuk hibah otorisasi yang paling aman. Itu tidak menampilkan konten token langsung ke pengguna Anda. Sebagai gantinya, aplikasi Anda bertanggung jawab untuk mengambil dan menyimpan token pengguna Anda dengan aman. Di Amazon Cognito, pemberian kode otorisasi adalah satu-satunya cara untuk mendapatkan ketiga jenis token—ID, akses, dan penyegaran—dari server otorisasi. Anda juga bisa mendapatkan ketiga jenis token dari otentikasi melalui API kumpulan pengguna Amazon Cognito, tetapi API tidak mengeluarkan token akses dengan cakupan selain `aws.cognito.signin.user.admin`

## Hibah implisit

Menanggapi permintaan otentikasi Anda yang berhasil, server otorisasi menambahkan token akses dalam `access_token` parameter, dan token ID dalam `id_token` parameter, ke URL panggilan balik Anda. Hibah implisit tidak memerlukan interaksi tambahan dengan [Titik akhir token](#). Untuk meminta hibah implisit, setel `response_type` ke token dalam permintaan Anda. Hibah implisit hanya menghasilkan ID dan token akses. Untuk contoh permintaan, lihat [Pemberian token tanpa cakupan openid](#).

Hibah implisit adalah hibah otorisasi warisan. Tidak seperti hibah kode otorisasi, pengguna dapat mencegat dan memeriksa token Anda. Untuk mencegah pengiriman token melalui hibah implisit, konfigurasi klien aplikasi Anda untuk mendukung pemberian kode otorisasi saja.

## Kredensial klien

Kredensial klien adalah hibah otorisasi khusus untuk akses. machine-to-machine Untuk menerima hibah kredensial klien, lewati [Otorisasi titik akhir](#) dan buat permintaan langsung ke [Titik akhir token](#). Klien aplikasi Anda harus memiliki rahasia klien dan hanya mendukung kredensial klien. Menanggapi permintaan Anda yang berhasil, server otorisasi mengembalikan token akses.



Token akses dari hibah kredensial klien adalah mekanisme otorisasi yang berisi cakupan OAuth 2.0. Biasanya, token berisi klaim cakupan khusus yang mengotorisasi operasi HTTP untuk mengakses API yang dilindungi. Untuk informasi selengkapnya, lihat [Cakupan, M2M, dan otorisasi API dengan server sumber daya](#).

Hibah kredensial klien menambah biaya ke tagihan Anda. AWS Untuk informasi selengkapnya, lihat [Harga Amazon Cognito](#).

## Menggunakan PKCE dalam hibah kode otorisasi dengan kumpulan pengguna Amazon Cognito

Amazon Cognito mendukung otentikasi Kunci Bukti untuk Pertukaran Kode (PKCE) dalam hibah kode otorisasi. PKCE adalah perpanjangan dari hibah kode otorisasi OAuth 2.0 untuk klien publik. PKCE menjaga terhadap penebusan kode otorisasi yang dicegat.

### Bagaimana Amazon Cognito menggunakan PKCE

Untuk memulai otentikasi dengan PKCE, aplikasi Anda harus menghasilkan nilai string yang unik. String ini adalah pemverifikasi kode, nilai rahasia yang digunakan Amazon Cognito untuk membandingkan klien yang meminta hibah otorisasi awal kepada klien yang menukar kode otorisasi dengan token.

Aplikasi Anda harus menerapkan hash SHA256 ke string pemverifikasi kode dan menyandikan hasilnya ke base64. Berikan string hash ke `code_challenge` parameter [Otorisasi titik akhir](#) as a di badan permintaan. Saat aplikasi Anda menukar kode otorisasi dengan token, kode tersebut harus menyertakan string pemverifikasi kode dalam teks biasa sebagai `code_verifier` parameter di badan permintaan ke file. [Titik akhir token](#) Amazon Cognito melakukan hash-and-encode operasi yang sama pada pemverifikasi kode. Amazon Cognito hanya mengembalikan ID, akses, dan token penyegaran jika menentukan bahwa pemverifikasi kode menghasilkan tantangan kode yang sama dengan yang diterimanya dalam permintaan otorisasi.

### Menerapkan Alur Hibah Otorisasi dengan PKCE

1. Buka Amazon Cognito [konsol](#). Jika diminta, masukkan AWS kredensial Anda.
2. Pilih Kolam Pengguna.
3. Pilih kolam pengguna yang ada dari daftar, atau [buat kolam pengguna](#). Jika Anda membuat kumpulan pengguna, Anda akan diminta untuk menyiapkan klien aplikasi dan mengonfigurasi UI yang dihosting selama wizard.

- a. Jika Anda membuat kumpulan pengguna baru, siapkan klien aplikasi dan konfigurasi UI yang dihosting selama persiapan terpandu.
  - b. Jika Anda mengonfigurasi kumpulan pengguna yang ada, tambahkan [domain](#) dan [klien aplikasi publik](#), jika Anda belum melakukannya.
4. Hasilkan string alfanumerik acak, biasanya pengidentifikasi unik universal (UUID), untuk membuat tantangan kode untuk PKCE. String ini adalah nilai `code_verifier` parameter yang akan Anda kirimkan dalam permintaan Anda ke [Titik akhir token](#).
  5. Hash `code_verifier` string dengan algoritma SHA256. Encode hasil operasi hashing ke base64. String ini adalah nilai `code_challenge` parameter yang akan Anda kirimkan dalam permintaan Anda ke [Otorisasi titik akhir](#).

PythonContoh berikut menghasilkan `code_verifier` dan menghitung: `code_challenge`

```
#!/usr/bin/env python3

import random
from base64 import urlsafe_b64encode
from hashlib import sha256
from string import ascii_letters
from string import digits

# use a cryptographically strong random number generator source
rand = random.SystemRandom()

code_verifier = ''.join(rand.choices(ascii_letters + digits, k=128))
code_verifier_hash = sha256(code_verifier.encode()).digest()
code_challenge = urlsafe_b64encode(code_verifier_hash).decode().rstrip('=')

print(f"code challenge: {code_challenge}")
print(f"code verifier: {code_verifier}")
```

Berikut ini adalah contoh output dari Python script:

```
code challenge: Eh0mg-0Zv7BAyo-tdv_vYamx1bo0YDu1DklyXoMDtLg
code verifier: 9D-aW_iygXrgQcWJd0y0tNVMPsXSchIc2xceDhvYVdGLCBk-
JWFTmBNjvKSd0rjTTYaz0FbUmrFERrjWx6oKtK2b6z_x4_gHBD1r4K1mRFgyE8yA-05-_v7Dxf3EIYJH
```

6. Selesaikan login UI yang dihosting dengan permintaan hibah kode otorisasi dengan PKCE. Berikut ini adalah contoh URL:

```
https://mydomain.us-east-1.amazoncognito.com/oauth2/authorize?
response_type=code&client_id=1example23456789&redirect_uri=https://
www.example.com&code_challenge=Eh0mg-0Zv7BAyo-
tdv_vYamx1bo0YDuLDklyXoMDtLg&code_challenge_method=S256
```

7. Kumpulkan otorisasi code dan tukarkan dengan token dengan titik akhir token. Berikut ini adalah contoh permintaan:

```
POST /oauth2/token HTTP/1.1
Host: mydomain.us-east-1.amazoncognito.com
Content-Type: application/x-www-form-urlencoded
Content-Length: 296

redirect_uri=https%3A%2F%2Fwww.example.com&
client_id=1example23456789&
code=7378f445-c87f-400c-855e-0297d072ff03&
grant_type=authorization_code&
code_verifier=9D-aW_iygXrgQcWJd0y0tNVMPsXSchIc2xceDhvYVdGLCBk-
JWFTmBNjvKSd0rjTTYaz0FbUmrFERrjWx6oKtK2b6z_x4_gHBDlr4K1mRFgyE8yA-05-_v7Dxf3EIYJH
```

8. Tinjau tanggapannya. Ini akan berisi ID, akses, dan token penyegaran. Untuk informasi selengkapnya tentang penggunaan token kumpulan pengguna Amazon Cognito, lihat [Menggunakan token dengan kolam pengguna](#)

## Respons kesalahan UI dan federasi yang dihosting

Proses masuk di UI yang dihosting atau login gabungan mungkin menampilkan kesalahan. Berikut ini adalah beberapa kondisi yang dapat menyebabkan otentikasi berakhir dengan kesalahan.

- Pengguna melakukan operasi yang tidak dapat dipenuhi oleh kumpulan pengguna Anda.
- Pemicu Lambda tidak merespons dengan sintaks yang diharapkan.
- Penyedia identitas Anda (iDP) mengembalikan kesalahan.
- Amazon Cognito tidak dapat memvalidasi informasi atribut yang diberikan pengguna Anda.
- IDP Anda tidak mengirim klaim yang memetakan ke atribut yang diperlukan.

Ketika Amazon Cognito menemukan kesalahan, ia mengkomunikasikannya dengan salah satu cara berikut.

1. Amazon Cognito mengirimkan URL pengalihan dengan kesalahan dalam parameter permintaan.
2. Amazon Cognito menampilkan kesalahan di UI yang dihosting.

Kesalahan yang ditambahkan Amazon Cognito ke parameter permintaan memiliki format berikut.

```
https://<Callback URL>/?error_description=error+description&error=error+name
```

Ketika Anda membantu pengguna Anda mengirimkan informasi kesalahan ketika mereka tidak dapat melakukan operasi, minta mereka menangkap URL dan teks atau tangkapan layar halaman.

#### Note

Deskripsi kesalahan Amazon Cognito bukanlah string tetap dan Anda tidak boleh menggunakan logika yang bergantung pada pola atau format tetap.

## Pesan kesalahan OIDC dan penyedia identitas sosial

Penyedia identitas Anda mungkin mengembalikan kesalahan. Jika IDP OIDC atau OAuth 2.0 menampilkan kesalahan yang sesuai dengan standar, Amazon Cognito mengalihkan pengguna Anda ke URL panggilan balik dan menambahkan respons kesalahan penyedia ke parameter permintaan kesalahan. Amazon Cognito menambahkan nama penyedia dan kode kesalahan HTTP ke string kesalahan yang ada.

URL berikut adalah contoh pengalihan dari iDP yang mengembalikan kesalahan ke Amazon Cognito.

```
https://www.amazon.com/?error_description=LoginWithAmazon+Error+-+400+invalid_request+The+request+is+missing+a+required+parameter+%3A+client_secret&error=invalid_request
```

Karena Amazon Cognito hanya mengembalikan apa yang diterimanya dari penyedia, pengguna Anda mungkin melihat subset informasi ini.

Ketika pengguna Anda mengalami masalah dengan login awal melalui iDP Anda, iDP mengirimkan pesan kesalahan langsung ke pengguna Anda. Amazon Cognito menyampaikan pesan kesalahan ke pengguna Anda saat membuat permintaan ke IDP Anda untuk memvalidasi sesi pengguna Anda. Amazon Cognito menyampaikan pesan kesalahan OAuth dan OIDC iDP dari titik akhir berikut.

/token

Amazon Cognito menukar kode otorisasi IDP dengan token akses.

/.well-known/openid-configuration

Amazon Cognito menemukan jalur ke titik akhir penerbit Anda.

/.well-known/jwks.json

Untuk memverifikasi Token Web JSON (JWT) pengguna Anda, Amazon Cognito menemukan JSON Web Keys (JWK) yang digunakan IDP Anda untuk menandatangani token.

Karena Amazon Cognito tidak memulai sesi keluar ke penyedia SAMP 2.0 yang mungkin menampilkan kesalahan HTTP, kesalahan pengguna Anda selama sesi dengan IDP SAMP 2.0 tidak menyertakan bentuk pesan kesalahan penyedia ini.

## Referensi API kumpulan pengguna Amazon Cognito

Dengan kumpulan pengguna Amazon Cognito, Anda dapat mendaftar dan masuk pengguna dengan web dan aplikasi seluler Anda. Anda dapat mengubah kata sandi untuk pengguna terautentikasi dan memulai alur kata sandi yang terlupakan untuk pengguna yang tidak terautentikasi. Untuk informasi lebih lanjut, lihat [Alur otentikasi kumpulan pengguna](#) dan [Menggunakan token dengan kolam pengguna](#).

API kumpulan pengguna Amazon Cognito mencakup operasi untuk melihat dan memodifikasi kumpulan pengguna dan pengguna Anda, serta untuk melakukan autentikasi dan otorisasi pengguna. Untuk deskripsi kelas operasi API yang digabungkan ke dalam API kumpulan pengguna Amazon Cognito, lihat [Menggunakan API kumpulan pengguna Amazon Cognito dan titik akhir kumpulan pengguna](#)

Untuk daftar terperinci operasi dan sintaks API kumpulan pengguna Amazon Cognito, lihat Referensi API kumpulan pengguna [Amazon Cognito](#). Setiap halaman di pengguna Amazon Cognito mengumpulkan tautan referensi API ke materi referensi dengan sintaks dan contoh untuk berbagai SDK. AWS

## Referensi API kumpulan identitas Amazon Cognito (identitas gabungan)

Dengan kolam identitas Amazon Cognito, pengguna web dan aplikasi seluler Anda dapat memperoleh kredensial sementara AWS dan hak istimewa terbatas yang memungkinkan mereka mengakses layanan AWS lain.

Untuk referensi API kumpulan identitas (identitas gabungan) lengkap, lihat Referensi API [Amazon Cognito](#).

## Referensi API sinkronisasi Amazon Cognito

Sinkronisasi Amazon Cognito adalah sebuah layanan AWS dan perpustakaan klien yang memungkinkan sinkronisasi lintas-perangkat data pengguna terkait aplikasi.

Untuk informasi selengkapnya tentang Referensi API Sinkronisasi Amazon Cognito, lihat [Referensi API Sinkronisasi Amazon Cognito](#).

## Riwayat dokumen untuk Amazon Cognito

Tabel berikut menjelaskan penambahan penting pada dokumentasi untuk Amazon Cognito. Kami juga sering melakukan pembaruan kecil pada dokumentasi sebagai tanggapan atas umpan balik yang Anda kirim. Untuk mengirimkan umpan balik, cari tautan Umpan Balik di bagian bawah halaman mana pun dalam dokumentasi Amazon Cognito.

Perubahan	Deskripsi	Tanggal
<a href="#">Menambahkan dukungan untuk objek kompleks di pemicu Lambda pra token</a>	Anda sekarang dapat menambahkan array dan objek JSON ke ID dan mengakses klaim token.	30 Mei 2024
<a href="#">Informasi terbaru tentang Izin Terverifikasi dan Amazon Cognito.</a>	Izin Terverifikasi Amazon sekarang memiliki lebih banyak integrasi langsung dengan Amazon Cognito.	15 Mei 2024
<a href="#">Identitas terverifikasi Amazon SES Multi-Wilayah.</a>	Di beberapa Wilayah AWS tanpa Amazon SES, kumpulan pengguna Amazon Cognito memuat saldo email antara dua Wilayah terpencil.	10 Mei 2024
<a href="#">Menambahkan informasi tentang otorisasi M2M dan mengelola biaya.</a>	Pelajari cara menggunakan hibah kredensial klien untuk kasus penggunaan machine-to-machine (M2M) dengan kumpulan pengguna Amazon Cognito.	9 Mei 2024
<a href="#">Amazon Cognito sekarang tersedia di Eropa (Spanyol) dan Asia Pasifik (Hyderabad). Wilayah AWS</a>	Anda sekarang dapat membuat sumber daya Amazon Cognito di Wilayah Eropa (Spanyol) dan Asia Pasifik (Hyderabad).	April 15, 2024

<a href="#">Amazon Cognito sekarang tersedia di Asia Pasifik (Melbourne). Wilayah AWS</a>	Anda sekarang dapat membuat sumber daya Amazon Cognito di Wilayah Asia Pasifik (Melbourne).	April 4, 2024
<a href="#">Menambahkan contoh aplikasi Android di kumpulan pengguna Flutter untuk Amazon Cognito.</a>	Anda dapat membuat aplikasi seluler pemula untuk Amazon Cognito dari contoh aplikasi Flutter di. GitHub	April 4, 2024
<a href="#">Konten baru memulai</a>	Konten yang diperluas untuk memulai, skenario umum, praktik terbaik multi-penyewa, dan mengakses sumber daya setelah masuk.	April 1, 2024
<a href="#">Amazon Cognito sekarang tersedia di Eropa (Zurich). Wilayah AWS</a>	Anda sekarang dapat membuat sumber daya Amazon Cognito di Wilayah Eropa (Zurich).	Maret 14, 2024
<a href="#">Amazon Cognito sekarang tersedia di Timur Tengah (UEA). Wilayah AWS</a>	Anda sekarang dapat membuat sumber daya Amazon Cognito di Wilayah Timur Tengah (UEA).	8 Maret 2024
<a href="#">Fitur SAFL baru dan konten yang ditingkatkan.</a>	Anda sekarang dapat menandatangani permintaan SAMP, mengenkripsi respons SAMP, dan menyiapkan SAMP SSO yang diprakarsai IDP.	Februari 1, 2024
<a href="#">Kuota meningkat tersedia.</a>	Anda sekarang dapat membeli kapasitas tambahan untuk kuota tingkat permintaan Amazon Cognito.	Januari 25, 2024



<a href="#">Kumpulan identitas Amazon Cognito mendukung tarif permintaan dalam Service Quotas.</a>	Anda sekarang dapat memantau kuota requests-per-second (RPS) untuk kumpulan identitas Amazon Cognito dan meminta peningkatan di konsol Service Quotas.	Desember 19, 2023
<a href="#">Ditambahkan fitur baru untuk kustomisasi isi token akses.</a>	Anda sekarang dapat menambahkan, memodifikasi, dan menghapus klaim dan cakupan dalam token akses kumpulan pengguna.	Desember 12, 2023
<a href="#">Konten yang disempurnakan tentang klien aplikasi dan cakupan OAuth.</a>	Pengeditan kejelasan dan koreksi ke <a href="#">Klien aplikasi kumpulan pengguna</a> dan <a href="#">Cakupan, M2M, dan otorisasi API dengan server sumber daya</a> Instruksi konsol lama yang dihapus.	14 November 2023
<a href="#">Konten yang disempurnakan tentang perangkat dan otentikasi perangkat.</a>	Konten baru tentang penggunaan kunci perangkat dan otentikasi SRP perangkat.	18 Oktober 2023
<a href="#">AWS Management Console Panduan yang diperbarui.</a>	Menghapus referensi konsol kumpulan pengguna dan topik yang didistribusikan ulang dalam subjek terkait, dan menambahkan panduan ke organisasi berbasis tab di konsol Amazon Cognito.	Agustus 30, 2023

<a href="#">Akses langsung yang tidak ditekankan ke titik akhir LOGIN.</a>	Menambahkan gambaran visual dari kumpulan pengguna <a href="#">Titik akhir masuk</a> dan menekankan otentikasi awal dengan. <a href="#">Otorisasi titik akhir</a>	Agustus 30, 2023
<a href="#">Amazon Cognito sekarang tersedia di Asia Pasifik (Osaka) dan Israel (Tel Aviv). Wilayah AWS</a>	Anda sekarang dapat membuat sumber daya Amazon Cognito di Wilayah Asia Pasifik (Osaka) dan Israel (Tel Aviv).	Agustus 30, 2023
<a href="#">Memperkenalkan informasi tentang otorisasi untuk Amazon Cognito dengan Izin Terverifikasi Amazon.</a>	Di aplikasi, Anda dapat memanggil API Izin Terverifikasi untuk menghasilkan keputusan akses dari otoritas pusat.	1 Agustus 2023
<a href="#">Menambahkan fitur baru untuk mencatat aktivitas pengguna terperinci kumpulan pengguna ke Amazon CloudWatch Logs.</a>	Anda sekarang dapat mencatat kesalahan pengiriman email dan pesan SMS ke grup CloudWatch log.	1 Agustus 2023
<a href="#">Informasi terbaru tentang kebijakan AWS terkelola untuk pengguna tamu kumpulan identitas.</a>	Cakupan izin untuk pengguna tamu kumpulan identitas sekarang mencakup kebijakan sesi sebaris dan kebijakan sesi terkelola. AWS	16 Mei 2023
<a href="#">Peningkatan konten dan instruksi konsol baru untuk kumpulan identitas Amazon Cognito.</a>	Menambahkan penelusuran konsol baru untuk mencerminkan pengalaman konsol baru, detail integrasi kode yang ditingkatkan untuk kumpulan identitas.	16 Mei 2023

<a href="#">Penambahan dan peningkatan pada beranda layanan dan beranda kumpulan pengguna.</a>	Halaman ikhtisar yang diperbarui untuk Amazon Cognito dan kumpulan <a href="#">pengguna</a> .	16 Mei 2023
<a href="#">Perbaikan umum pada dokumentasi token kumpulan pengguna.</a>	Token contoh yang diperbarui, menambahkan informasi baru tentang memverifikasi token.	16 Februari 2023
<a href="#">Anda sekarang dapat mencatat peristiwa data kumpulan identitas Amazon Cognito. AWS CloudTrail</a>	CloudTrail mendukung pemilihan identitas Amazon Cognito mengumpulkan operasi API volume tinggi di jalur yang mencatat peristiwa data.	15 Februari 2023
<a href="#">Contoh dan deskripsi pemicu Lambda yang diperbarui.</a>	Contoh pemicu Lambda diperbarui ke JavaScript versi 3. Anda sekarang dapat secara langsung menghubungkan pemicu Lambda dengan tindakan API.	31 Januari 2023
<a href="#">Kumpulan identitas Amazon Cognito menerapkan kebijakan AWS terkelola ke sesi yang tidak diautentikasi.</a>	Pengguna kumpulan identitas yang melakukan autentikasi menggunakan alur yang disempurnakan sekarang memiliki kebijakan AWS terkelola tambahan yang diterapkan pada sesi mereka.	31 Januari 2023
<a href="#">Contoh kode yang ditambahkan.</a>	Panduan ini sekarang menyertakan kode contoh untuk aplikasi Amazon Cognito Anda dalam berbagai bahasa pemrograman.	23 Januari 2023

<a href="#">Menambahkan informasi tentang model API dan otentikasi dengan kumpulan pengguna Amazon Cognito.</a>	Kumpulan pengguna Amazon Cognito memiliki beberapa antarmuka dan format API untuk otorisasi permintaan.	Desember 15, 2022
<a href="#">Amazon Cognito sekarang tersedia di Eropa (Milan). Wilayah AWS</a>	Anda sekarang dapat membuat kumpulan pengguna Amazon Cognito di Wilayah Eropa (Milan).	6 Desember 2022
<a href="#">Menambahkan informasi tentang perlindungan penghapusan kumpulan pengguna.</a>	Saat Anda membuat kumpulan pengguna baru dengan AWS Management Console, sekarang dilindungi dari penghapusan secara default.	20 Oktober 2022
<a href="#">Menambahkan panduan pengguna untuk UI yang dihosting, dan informasi tentang TOTP MFA di UI yang dihosting.</a>	Pengguna Anda sekarang dapat mendaftarkan perangkat MFA TOTP di UI yang dihosting Amazon Cognito. Anda sekarang dapat melihat pratinjau UI yang di-host default.	8 September 2022
<a href="#">Menambahkan informasi tentang AWS WAF dan Amazon Cognito.</a>	Anda sekarang dapat mengaitkan ACL AWS WAF web dengan kumpulan pengguna Amazon Cognito.	3 Agustus 2022
<a href="#">Ditambahkan lebih banyak contoh AWS CloudTrail acara.</a>	Amazon Cognito sekarang mencatat federasi dan menghosting permintaan UI ke jejak Anda.	15 Juni 2022

[Menambahkan informasi tentang verifikasi atribut dua langkah.](#)

Anda sekarang dapat memilih apakah pengguna Anda harus memverifikasi alamat email atau nomor telepon baru sebelum mereka dapat masuk dengannya.

9 Juni 2022

[Diperbarui dokumentasi federasi. Fitur propagasi alamat IP baru.](#)

Panduan yang diperbarui untuk menyiapkan kumpulan sosial pengguna. IdPs Menambahkan informasi tentang profil pengguna federasi dan pemetaan atribut. Menambahkan informasi baru tentang sidik jari perangkat untuk keamanan tingkat lanjut.

31 Mei 2022

[Masuk pengguna federasi tanpa interaksi dengan UI yang dihosting](#)

Menambahkan halaman baru tentang cara mem-bookmark aplikasi sehingga Amazon Cognito diam-diam mengarahkan pengguna ke login federasi.

Mei 29, 2022

[Pesan SMS dan email di Wilayah untuk kumpulan pengguna Amazon Cognito](#)

Anda sekarang dapat menggunakan Amazon Simple Notification Service untuk pesan SMS dan Amazon Simple Email Service untuk pesan email yang Wilayah AWS sama dengan kumpulan pengguna Anda.

Maret 14, 2022

[Pembaruan ke halaman kuota](#)

Menambahkan dan mengklarifikasi kuota sumber daya dan tingkat permintaan.

10 Januari 2022

---

<a href="#">Pengalaman konsol kumpulan pengguna Amazon Cognito baru</a>	Instruksi yang diperbarui untuk membuat dan mengelola kumpulan pengguna di konsol Amazon Cognito yang diperbarui.	18 November 2021
<a href="#">RevokeToken API dan Titik Akhir Pencabutan</a>	Anda dapat menggunakan RevokeToken operasi untuk <a href="#">mencabut token penyegaran</a> bagi pengguna.	10 Juni 2021
<a href="#">Praktik terbaik multi-penyewa</a>	Menambahkan praktik terbaik untuk aplikasi multi-penyewa.	4 Maret 2021
<a href="#">Atribut untuk kontrol akses</a>	Amazon Cognito Identity Pools menyediakan atribut untuk kontrol akses (AFAC) sebagai cara bagi pelanggan untuk memberi pengguna akses ke sumber daya. AWS Otorisasi dapat dilakukan berdasarkan atribut pengguna dari penyedia identitas yang mereka gunakan untuk bergabung dengan Amazon Cognito.	15 Januari 2021
<a href="#">Pemicu Lambda Pengirim SMS Kustom dan Pemicu Lambda Pengirim Email Kustom</a>	Pemicu Lambda Pengirim SMS Kustom dan Pemicu Lambda Pengirim Email Kustom memungkinkan Anda mengaktifkan penyedia pihak ketiga untuk mengirim notifikasi email dan SMS ke pengguna Anda dari dalam kode fungsi Lambda Anda.	30 November 2020

---

<a href="#">Pembaruan token Amazon Cognito</a>	Informasi kedaluwarsa yang diperbarui telah ditambahkan ke Akses, ID, dan token Refresh.	29 Oktober 2020
<a href="#">Service Quotas Amazon Cognito</a>	Service Quotas tersedia untuk Amazon Cognito kategori kuota. Anda dapat menggunakan konsol Service Quotas untuk melihat penggunaan kuota, meminta peningkatan kuota, dan membuat CloudWatch alarm untuk memantau penggunaan kuota Anda. Sebagai bagian dari perubahan ini, bagian CloudWatch Metrik Tersedia untuk Kumpulan Pengguna Amazon Cognito diperbarui untuk mencerminkan informasi baru. Nama bagian baru adalah: Melacak kuota dan penggunaan di CloudWatch dan Service Quotas	29 Oktober 2020
<a href="#">Kategorisasi kuota Amazon Cognito</a>	Kategori kuota tersedia untuk membantu Anda memantau penggunaan kuota dan meminta peningkatan. Kuota dikelompokkan ke dalam kategori berdasarkan kasus penggunaan umum.	17 Agustus 2020
<a href="#">Amazon Cognito didukung di GovCloud AS AWS</a>	Amazon Cognito sekarang didukung di Wilayah AWS GovCloud (AS).	13 Mei 2020

---

<a href="#">Pembaruan dokumen Amazon Cognito Pinpoint</a>	Peran terkait layanan baru telah ditambahkan. Petunjuk telah diperbarui pada "Menggunakan Amazon Pinpoint Analytics dengan Kolam Pengguna Amazon Cognito".	13 Mei 2020
<a href="#">Bab keamanan khusus Amazon Cognito baru</a>	Bab Keamanan dapat membantu organisasi Anda mendapatkan informasi mendalam tentang keamanan layanan bawaan dan yang dapat dikonfigurasi. AWS Bab baru kami memberikan informasi tentang keamanan cloud dan di cloud.	30 April 2020
<a href="#">Amazon Cognito Identity Pools sekarang mendukung Masuk dengan Apple</a>	Masuk dengan Apple tersedia di semua wilayah tempat Amazon Cognito beroperasi, kecuali wilayah cn-north-1.	7 April 2020
<a href="#">Versi API Facebook Baru</a>	Menambahkan pilihan versi ke Facebook API.	3 April 2020
<a href="#">Pembaruan ketidakpekaan kasus nama pengguna</a>	Menambahkan rekomendasi tentang mengaktifkan ketidakpekaan huruf besar-kecil nama pengguna sebelum membuat kolam pengguna.	11 Februari 2020



[Informasi baru tentang AWS Amplify](#)

Menambahkan informasi tentang mengintegrasikan Amazon Cognito dengan web atau aplikasi seluler Anda dengan AWS Amplify menggunakan SDK dan pustaka. Menghilangkan informasi tentang penggunaan SDK Amazon Cognito yang mendahului AWS Amplify.

22 November 2019

[Atribut baru untuk pemicu kumpulan pengguna](#)

Amazon Cognito sekarang menyertakan `clientMetadata` parameter dalam informasi peristiwa yang diteruskan ke AWS Lambda fungsi untuk sebagian besar pemicu kumpulan pengguna. Anda dapat menggunakan parameter ini untuk meningkatkan alur kerja autentikasi kustom Anda dengan data tambahan.

4 Oktober 2019

[Batas yang diperbarui](#)

Batas pembatasan untuk tindakan `ListUsers` API diperbarui.

25 Juni 2019

[Batas baru](#)

Batas lunak untuk kolom pengguna sekarang mencakup batas untuk jumlah pengguna.

17 Juni 2019

---

<a href="#">Pengaturan email Amazon SES untuk kumpulan pengguna Amazon Cognito</a>	Anda dapat mengonfigurasi kolam pengguna sehingga Amazon Cognito mengirim email kepada pengguna Anda dengan menggunakan konfigurasi Amazon SES Anda. Pengaturan ini memungkinkan Amazon Cognito untuk mengirim email dengan volume pengiriman yang lebih tinggi daripada yang dimungkinkan.	8 April 2019
<a href="#">Dukungan penandaan</a>	Menambahkan informasi tentang menandai sumber daya Amazon Cognito.	26 Maret 2019
<a href="#">Mengubah sertifikat untuk domain kustom</a>	Jika Anda menggunakan domain kustom untuk menghosting UI yang dihosting Amazon Cognito, Anda dapat mengubah sertifikat SSL untuk domain ini sesuai kebutuhan.	19 Desember 2018
<a href="#">Batas baru</a>	Batas baru ditambahkan untuk jumlah maksimum grup yang dapat dimiliki oleh setiap pengguna.	14 Desember 2018
<a href="#">Batas yang diperbarui</a>	Batas lunak untuk kolam pengguna diperbarui.	11 Desember 2018

---

<a href="#">Pembaruan dokumentasi untuk memverifikasi alamat email dan nomor telepon</a>	Menambahkan informasi tentang mengonfigurasi kolam pengguna Anda untuk meminta verifikasi email atau telepon saat pengguna mendaftar di aplikasi Anda.	20 November 2018
<a href="#">Pembaruan dokumentasi untuk menguji email</a>	Menambahkan panduan untuk memulai email dari Amazon Cognito saat Anda menguji aplikasi Anda.	13 November 2018
<a href="#">Keamanan Lanjutan Amazon Cognito</a>	Menambahkan fitur keamanan baru untuk memungkinkan developer melindungi aplikasi dan pengguna mereka dari bot berbahaya, mengamankan akun pengguna dari kredensial yang disusupi, dan secara otomatis menyesuaikan tantangan yang diperlukan untuk masuk berdasarkan risiko yang diperhitungkan dari upaya masuk.	14 Juni 2018
<a href="#">Domain Kustom untuk UI yang Dihosting Amazon Cognito</a>	Mengizinkan developer untuk menggunakan domain kustom sepenuhnya mereka sendiri untuk UI yang dihosting di Kolam Pengguna Amazon Cognito.	4 Juni 2018
<a href="#">Penyedia Identitas OIDC Kumpulan Pengguna Amazon Cognito</a>	Menambahkan masuk kolam pengguna melalui penyedia identitas OpenID Connect (OIDC) seperti Salesforce atau Ping Identity.	17 Mei 2018

---

<a href="#">Pemicu Migrasi Amazon Cognito Lambda</a>	Menambahkan halaman yang mencakup fitur Pemicu Migrasi Lambda	April 8, 2018
<a href="#">Pembaruan Panduan Pengembang Amazon Cognito</a>	Menambahkan tingkat atas “Apa itu Amazon Cognito” dan “Memulai dengan Amazon Cognito”. Juga menambahkan skenario umum dan mengatur ulang TOC kolom pengguna. Menambahkan bagian “Memulai dengan kolom pengguna Amazon Cognito” baru.	6 April 2018
<a href="#">Amazon Cognito Advanced Security Beta</a>	Menambahkan fitur keamanan baru untuk memungkinkan developer melindungi aplikasi dan pengguna mereka dari bot berbahaya, mengamankan akun pengguna terhadap kredensial di tempat liar yang telah disusupi di tempat lain di internet, dan secara otomatis menyesuaikan tantangan yang diperlukan untuk masuk berdasarkan risiko yang diperhitungkan dari upaya masuk.	28 November 2017

[Integrasi Amazon Pinpoint](#)

Menambahkan kemampuan untuk menggunakan Amazon Pinpoint untuk menyediakan analitik bagi aplikasi Kolam Pengguna Amazon Cognito Anda dan untuk memperkaya data pengguna untuk kampanye Amazon Pinpoint.

26 September 2017

[Federasi dan fitur UI aplikasi bawaan dari kumpulan pengguna Amazon Cognito](#)

Menambahkan kemampuan untuk mengizinkan pengguna Anda masuk ke kolam pengguna Anda melalui Facebook, Google, Login with Amazon, atau penyedia identitas SAML. Menambahkan UI aplikasi bawaan yang dapat disesuaikan dan dukungan OAuth 2.0 dengan klaim kustom.

10 Agustus 2017

[Perubahan fitur terkait kepatuhan HIPAA dan PCI](#)

Menambahkan kemampuan untuk mengizinkan pengguna Anda menggunakan nomor telepon atau alamat email sebagai nama pengguna mereka.

6 Juli 2017

[Grup pengguna dan fitur kontrol akses berbasis peran](#)

Menambahkan kemampuan administratif untuk membuat dan mengelola grup pengguna. Administrator dapat menetapkan IAM role kepada pengguna berdasarkan keanggotaan grup dan aturan yang dibuat administrator.

15 Desember 2016

---

<a href="#">Pembaruan dokumentasi</a>	Contoh terbaru yang menunjukkan cara menggunakan AWS Lambda pemicu dengan kumpulan pengguna.	November 27, 2016
<a href="#">Pembaruan dokumentasi</a>	Contoh kode iOS yang diperbarui.	18 November 2016
<a href="#">Pembaruan dokumentasi</a>	Menambahkan informasi tentang alur konfirmasi untuk akun pengguna.	9 November 2016
<a href="#">Buat fitur akun pengguna</a>	Menambahkan kemampuan administratif untuk membuat akun pengguna melalui konsol Amazon Cognito dan API.	6 Oktober 2016
<a href="#">Fitur impor pengguna</a>	Menambahkan kemampuan impor massal untuk Kolam Pengguna Cognito. Gunakan fitur ini untuk memigrasi pengguna dari penyedia identitas yang ada ke kolam pengguna Amazon Cognito.	1 September 2016
<a href="#">Ketersediaan umum Cognito User Pools</a>	Menambahkan fitur Kolam Pengguna Cognito. Gunakan fitur ini untuk membuat dan memelihara direktori pengguna dan menambahkan pendaftaran dan masuk ke aplikasi seluler atau aplikasi web Anda menggunakan kolam pengguna.	28 Juli 2016

---

<a href="#">Dukungan SAFL</a>	Menambahkan dukungan untuk autentikasi dengan penyedia identitas melalui Security Assertion Markup Language 2.0 (SAML 2.0).	23 Juni 2016
<a href="#">CloudTrail integrasi</a>	Menambahkan integrasi dengan AWS CloudTrail.	18 Februari 2016
<a href="#">Integrasi acara dengan Lambda</a>	Memungkinkan Anda menjalankan AWS Lambda fungsi sebagai respons terhadap peristiwa penting di Amazon Cognito.	9 April 2015
<a href="#">Aliran data ke Amazon Kinesis</a>	Memberikan kontrol dan wawasan tentang aliran data Anda.	4 Maret 2015
<a href="#">Dukungan OpenID Connect</a>	Memungkinkan dukungan untuk penyedia OpenID Connect.	November 23, 2014
<a href="#">Sinkronisasi dorong</a>	Memungkinkan dukungan untuk sinkronisasi push senyap.	6 November 2014
<a href="#">Dukungan identitas yang diautentikasi pengembang ditambahkan</a>	Memungkinkan developer yang memiliki sistem autentikasi dan manajemen identitas mereka sendiri untuk diperlakukan sebagai penyedia identitas di Amazon Cognito.	29 September 2014
<a href="#">Ketersediaan umum Amazon Cognito</a>		10 Juli 2014

Terjemahan disediakan oleh mesin penerjemah. Jika konten terjemahan yang diberikan bertentangan dengan versi bahasa Inggris aslinya, utamakan versi bahasa Inggris.