aws

Panduan Pengguna

AWS Secrets Manager



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

AWS Secrets Manager: Panduan Pengguna

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

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

Table of Contents

Apa itu Manajer Rahasia? 1
Memulai dengan Secrets Manager 1
Kepatuhan dengan standar 2
Harga 2
Mengakses Secrets Manager 4
Konsol Manajer Rahasia 4
Alat baris perintah 4
AWS SDK 5
API Kueri HTTPS 5
Titik akhir Manajer Rahasia 6
Praktik terbaik 12
Menyimpan kredensi dan informasi sensitif lainnya di AWS Secrets Manager 12
Temukan rahasia yang tidak dilindungi dalam kode Anda 13
Pilih kunci enkripsi untuk rahasia Anda 13
Gunakan caching untuk mengambil rahasia 13
Putar rahasia Anda 14
Mengurangi risiko menggunakan CLI 14
Batasi akses ke rahasia 14
`BlockPublicPolicy`kondisi 15
Berhati-hatilah dengan kondisi alamat IP dalam kebijakan 15
Batasi permintaan dengan kondisi titik akhir VPC 16
Replikasi rahasia 16
Memantau rahasia 17
Jalankan infrastruktur Anda di jaringan pribadi 17
Kunci API untuk beban kerja sensitif 18
Simpan kunci API di AWS Secrets Manager 18
Konfigurasi enkripsi 19
Rotasi otomatis untuk kunci API 20
Siklus hidup fungsi rotasi 21
Konfigurasi jadwal rotasi 22
Mengambil rahasia secara efisien 22
Pengerasan keamanan untuk beban kerja sensitif 23
Pertanyaan umum 24
Bagaimana cara menangani periode tumpang tindih selama rotasi? 24

Bagaimana jika penyedia saya tidak mendukung pembuatan kunci terprogram? 24
Bagaimana cara menghindari pelambatan API saat mengambil rahasia? 24
Haruskah saya menggunakan satu rahasia per lingkungan atau satu rahasia dengan versi? 25
Tutorial 26
CodeGuru Peninjau Amazon 26
Ganti rahasia hardcode 26
Langkah 1: Buat Rahasianya 27
Langkah 2: Perbarui kode Anda 29
Langkah 3: Perbarui rahasianya 30
Langkah selanjutnya 30
Ganti kredensi DB hardcode 31
Langkah 1: Buat Rahasianya 31
Langkah 2: Perbarui kode Anda 33
Langkah 3: Putar rahasianya 33
Langkah selanjutnya 34
Buat rahasia klien Salesforce 35
Langkah 1: Buat Aplikasi Klien Eksternal dan aktifkan OAuth 36
Langkah 2: Buat rahasia eksternal yang dikelola dan aktifkan rotasi 37
Langkah 3: Uji kredensi yang diputar 38
Langkah 4: Bersihkan Sumber Daya 39
Langkah selanjutnya 40
Rotasi pengguna bergantian 40
Izin 41
Prasyarat 41
Langkah 1: Buat pengguna database Amazon RDS 44
Langkah 2: Buat rahasia untuk kredensi pengguna 47
Langkah 3: Uji rahasia yang diputar 48
Langkah 4: Bersihkan Sumber Daya 49
Langkah selanjutnya 50
Rotasi pengguna tunggal 50
Izin 50
Prasyarat 51
Langkah 1: Buat pengguna database Amazon RDS 51
Langkah 2: Buat rahasia untuk kredensi pengguna database 52
Langkah 3: Uji kata sandi yang diputar 53

Langkah 4: Bersihkan Sumber Daya 54
Langkah selanjutnya 54
Buat rahasia 55
AWS CLI 58
AWS SDK 59
Apa yang ada dalam rahasia 59
Metadata 59
Versi rahasia 60
Struktur JSON dari sebuah rahasia 62
Kredensi Amazon RDS dan Aurora 62
Kredensi Amazon Redshift 65
Kredensi Tanpa Server Amazon Redshift 65
Kredensi Amazon DocumentDB 66
Amazon Timestream untuk struktur rahasia InfluxDB 66
ElastiCache Kredensi Amazon 67
Kredensil Direktori Aktif 67
Kelola rahasia 69
Memperbarui nilai rahasia 69
AWS CLI 70
AWS SDK 70
Buat kata sandi dengan Secrets Manager 71
Kembalikan rahasia ke versi sebelumnya 71
Mengubah kunci enkripsi untuk rahasia 71
AWS CLI 73
Ubah rahasia 74
AWS CLI 75
AWS SDK 75
Temukan rahasia 76
Penelusuran filter 76
AWS CLI 77
AWS SDK 78
Hapus rahasia 78
AWS CLI 79
AWS SDK 80
Kembalikan rahasia 80
AWS CLI 81

AWS SDK 81
Kata kunci rahasia 82
Tinjau dasar-dasar tag 82
Lacak biaya menggunakan penandaan 83
Memahami batasan tag 83
Menandai rahasia di konsol 84
AWS CLI 85
API 86
SDK 86
Multi-region replikasi 87
AWS CLI 88
AWS SDK 89
Promosikan rahasia replika menjadi rahasia mandiri 89
AWS CLI 90
AWS SDK 90
Mencegah replikasi 90
Pecahkan masalah replikasi 92
Rahasia dengan nama yang sama ada di Wilayah yang dipilih 92
Tidak ada izin yang tersedia pada kunci KMS untuk menyelesaikan replikasi 92
Kunci KMS dinonaktifkan atau tidak ditemukan 93
Anda belum mengaktifkan Wilayah tempat replikasi terjadi 93
Dapatkan rahasia 94
Java 95
Java dengan caching sisi klien 95
Koneksi JDBC dengan kredensil secara rahasia 102
AWS SDK Jawa 111
Python 113
Python dengan caching sisi klien 114
SDK Python AWS 120
Dapatkan sejumlah nilai rahasia 122
.NET 123
.NET dengan caching sisi klien 124
SDK untuk .NET 130
Go 133
Pergi dengan caching sisi klien 134
Pergi AWS SDK 138

Karat 139
Karat dengan caching sisi klien 139
Karat 142
Amazon EKS 142
ASCP dengan Peran IAM untuk Akun Layanan (IRSA) 143
ASCP dengan Identitas Pod 143
Memilih pendekatan yang tepat 144
Instal ASCP untuk Amazon EKS 144
Integrasikan ASCP dengan Pod Identity untuk Amazon EKS 148
Integrasikan ASCP dengan IRSA untuk Amazon EKS 152
Contoh ASCP 155
AWS Lambda 163
Dapatkan rahasia dengan Lambda 163
Integrasi Parameter Store 164
AWS Penyedia Kredensia Beban Kerja 164
Bagaimana AWS Penyedia KredenSIAL Beban Kerja berfungsi 164
Memahami AWS Caching Penyedia KredenSIAL Beban Kerja 165
Unduh AWS Penyedia Kredensia Beban Kerja 166
Bangun AWS Penyedia KredenSIAL Beban Kerja dari sumber 167
Instal AWS Penyedia Kredensia Beban Kerja 170
Mengambil rahasia dengan AWS Penyedia Kredensia Beban Kerja 176
Memahami parameter `RefreshNow` 178
Mengambil rahasia di seluruh akun dengan rantai peran 181
Pre-fetch rahasia saat startup 183
Opsi konfigurasi 185
Fitur opsional 187
Pencatatan log 188
Pertimbangan keamanan 189
Agen Pengkodean AI 189
Cara kerjanya 190
Prasyarat 190
Pasang plugin 190
`{{resolve:...}}` sintaksis 191
Gunakan `asm-exec` untuk menjalankan perintah dengan rahasia 191
Cross-region rahasia 192
Pertimbangan keamanan 193

Bagaimana hook memblokir akses rahasia langsung 193
Pemecahan masalah 193
C++ 194
JavaScript 195
Kotlin 196
PHP 197
Ruby 198
AWS CLI 199
Dapatkan sekelompok rahasia dalam satu batch menggunakan AWS CLI 199
AWS konsol 200
AWS Batch 200
CloudFormation 201
GitHub Lowongan 202
Prasyarat 203
Penggunaan 203
Penamaan variabel lingkungan 204
Contoh 205
GitLab 208
Pertimbangan-pertimbangan 208
Prasyarat 208
Mengintegrasikan AWS Secrets Manager dengan GitLab 210
Pemecahan masalah 211
AWS IoT Greengrass 212
Penyimpanan Parameter 212
Merotasi rahasia 214
Rotasi terkelola 214
Putar rahasia eksternal yang dikelola 216
Mengatur Rotasi di Konsol 216
Mengatur Rotasi Menggunakan CLI 217
Rotasi dengan fungsi Lambda 218
Rotasi otomatis untuk rahasia database (konsol) 219
Rotasi otomatis untuk rahasia non-database (konsol) 223
Rotasi otomatis (AWS CLI) 228
Strategi rotasi fungsi Lambda 231
Fungsi rotasi Lambda 233
Templat fungsi rotasi 236

Izin untuk rotasi 244
Akses jaringan untuk fungsi AWS Lambda rotasi 249
Memecahkan masalah rotasi 250
Jadwal rotasi 268
Jendela rotasi 269
Ekspresi rate 269
Ekspresi Cron 269
Putar rahasia segera 275
AWS CLI 275
Temukan rahasia yang tidak diputar 276
Batalkan rotasi otomatis 276
Rahasia dikelola oleh layanan lain 278
Layanan yang menggunakan rahasia 279
App Runner 281
App2Container AWS 281
AWS AppConfig 281
Amazon AppFlow 282
AWS AppSync 282
Amazon Athena 282
Amazon Aurora 282
AWS CodeBuild 283
Amazon Data Firehose 283
AWS DataSync 283
Amazon DataZone 284
Direct Connect 284
AWS Directory Service 284
Amazon DocumentDB 285
AWS Elastic Beanstalk 285
Amazon Elastic Container Registry 285
Amazon Elastic Container Service 286
Amazon ElastiCache 286
AWS Elemental Live 287
AWS Elemental MediaConnect 287
AWS Elemental MediaConvert 287
AWS Elemental MediaLive 287
AWS Elemental MediaPackage 288

AWS Elemental MediaTailor 288
Amazon EMR 288
Amazon EventBridge 289
Amazon FSx 289
AWS Glue DataBrew 289
AWS Glue Studio 290
AWS IoT SiteWise 290
Amazon Kendra 290
Amazon Kinesis Video Streams 291
AWS Launch Wizard 291
Amazon Lookout for Metrics 291
Amazon Managed Grafana 292
AWS Managed Services 292
Amazon Managed Streaming untuk Apache Kafka 292
Amazon Managed Workflows for Apache Airflow 292
AWS Marketplace 293
AWS Migration Hub 293
AWS Layanan Komputasi Paralel 293
AWS ParallelCluster 294
Amazon Q 294
Amazon OpenSearch Ingestion 294
AWS OpsWorks for Chef Automate 295
Amazon Quick 295
Amazon RDS 295
Amazon Redshift 296
Editor kueri Amazon Redshift v2 296
Amazon SageMaker AI 297
AWS SCT 297
Amazon Timestream untuk InfluxDB 298
AWS Toolkit for JetBrains 298
AWS Transfer Family 298
AWS Wickr 299
Rahasia yang dikelola oleh aplikasi pihak ketiga 300
Fitur utama 300
Mitra Integrasi 301
Token Penyegaran ID Besar 302

Cisco Kunci API Platform Keamanan 303
Kunci Confluent Cloud API 305
Databricks Token Akses Pribadi 307
Databricks Rahasia Klien Utama Layanan OAuth 309
Kunci Admin Datadog 311
Kunci API Datadog 313
Kunci Aplikasi Datadog 315
GitLab Token Akses 316
Jenkins Token API 318
Pengguna Database MongoDB Atlas 320
Kredensi Akun Layanan MongoDB Atlas 322
Netskope Token API 324
Kunci API Dayung 325
Rahasia Klien Salesforce 327
Pasangan Kunci Kepingan Salju 329
Token Akses Terprogram Kepingan Salju 330
SonarQube Token 332
Twilio Kunci API 334
Twilio Token Auth 337
Twilio Kunci API Utama 338
Keamanan dan izin 340
Pantau dan pecahkan masalah 342
Migrasi rahasia yang ada 342
Pertimbangan dan batasan 343
CloudFormation 344
Buat rahasia 344
JSON 345
YAML 345
Buat rahasia dengan kredensil Amazon RDS dengan rotasi otomatis 346
Buat rahasia dengan kredensi Amazon Redshift 346
Buat rahasia dengan kredensi Amazon DocumentDB 346
JSON 347
YAML 351
Bagaimana Secrets Manager menggunakan CloudFormation 353
AWS CDK 354
Pemberitahuan acara rahasia 355

Cara EventBridge merutekan acara Secrets Manager 355
Acara Secrets Manager 355
Struktur peristiwa 356
Membuat pola peristiwa 356
Label Rahasia Acara yang diperbarui 358
Memantau rahasia 360
Masuk dengan AWS CloudTrail 360
AWS CLI 361
CloudTrail entri 361
Monitor dengan CloudWatch 367
CloudWatch alarm 367
Acara Match Secrets Manager dengan EventBridge 368
Cocokkan semua perubahan dengan rahasia tertentu 368
Cocokkan acara saat nilai rahasia berputar 369
Memantau rahasia yang dijadwalkan untuk dihapus 370
Langkah 1: Konfigurasikan pengiriman file CloudTrail log ke CloudWatch Log 371
Langkah 2: Buat CloudWatch alarm 371
Langkah 3: Uji CloudWatch alarm 372
Pantau rahasia untuk kepatuhan 373
Biaya Monitor Secrets Manager 374
Mendeteksi ancaman dengan GuardDuty 374
Validasi kepatuhan 375
Standar kepatuhan 375
Keamanan 378
Mengurangi risiko menggunakan AWS CLI untuk menyimpan rahasia Anda AWS Secrets Manager 379
Kontrol autentikasi dan akses 381
Referensi izin 382
Izin administrator Secrets Manager 382
Izin untuk mengakses rahasia 382
Izin untuk fungsi rotasi Lambda 382
Izin untuk kunci enkripsi 382
Izin untuk replikasi 383
Identity-based kebijakan 383
Resource-based kebijakan 390
Kontrol akses ke rahasia menggunakan tag 397

AWS kebijakan yang dikelola ... 399
Tentukan siapa yang memiliki izin untuk rahasia Anda ... 405
Cross-accountakses ... 406
On-premises akses ... 410
Perlindungan data di Secrets Manager ... 410
Enkripsi saat diam ... 411
Enkripsi saat bergerak ... 411
Inter-network privasi lalu lintas ... 412
Pengelolaan kunci enkripsi ... 412
Enkripsi rahasia dan dekripsi ... 412
Memilih AWS KMS kunci ... 413
Apa yang dienkripsi? ... 414
Proses enkripsi dan dekripsi ... 414
Izin untuk kunci KMS ... 415
Bagaimana Secrets Manager menggunakan kunci KMS Anda ... 416
Kebijakan utama dari Kunci yang dikelola AWS (`aws/secretsmanager`) ... 418
Konteks enkripsi Secrets Manager ... 420
Memantau interaksi Secrets Manager dengan AWS KMS ... 422
Keamanan infrastruktur ... 426
Titik akhir VPC (AWS PrivateLink) ... 426
Membuat kebijakan titik akhir ... 427
Subnet bersama ... 428
Akses IPv4 dan IPv6 ... 429
Apa itu IPv6? ... 429
Menggunakan kebijakan dual-stack ... 429
Menambahkan IPv6 ke kebijakan ... 430
Memverifikasi klien Anda mendukung IPv6 ... 432
Ketahanan ... 433
Post-quantum TLS ... 433
Pemecahan masalah ... 436
Pesan “Akses ditolak” ... 436
“Akses ditolak” untuk kredensi keamanan sementara ... 437
Perubahan yang saya buat tidak selalu langsung terlihat. ... 437
Kesalahan “Koneksi diatur ulang oleh rekan” ... 438
“Tidak dapat menghasilkan kunci data dengan kunci KMS asimetris” saat membuat rahasia ... 439

Sesi AWS CLI atau AWS Operasi SDK tidak dapat menemukan rahasia saya dari ARN sebagian 439
Rahasia ini dikelola oleh seorang AWS layanan, dan Anda harus menggunakan layanan itu untuk memperbaruinya. 440
Impor modul Python gagal saat menggunakan `Transform`: -2024-09-16 AWS::SecretsManager 440
Rahasia eksternal saya yang dikelola tidak berputar 440
Kuota 442
Kuota Secrets Manager 442
Tambahkan percobaan ulang ke aplikasi Anda 445
Riwayat dokumen 447
Pembaruan lebih awal 448
..... cdxlix

Apa itu AWS Secrets Manager?

AWS Secrets Manager membantu Anda mengelola, mengambil, dan memutar kredenSIAL database, kredenSIAL aplikasi, token OAuth, kunci API, dan rahasia lainnya sepanjang siklus hidupnya. Banyak AWS layanan menyimpan dan menggunakan rahasia di Manajer Rahasia.

Secrets Manager membantu Anda meningkatkan postur keamanan, karena Anda tidak lagi memerlukan kredensial hard-code dalam kode sumber aplikasi. Menyimpan kredenSIAL di Secrets Manager membantu menghindari kemungkinan kompromi oleh siapa saja yang dapat memeriksa aplikasi Anda atau komponen. Anda menggantikan kredensial hard-code dengan panggilan runtime ke layanan Secrets Manager untuk mengambil kredensial secara dinamis saat Anda membutuhkannya.

Dengan Secrets Manager, Anda dapat mengonfigurasi jadwal rotasi otomatis untuk rahasia Anda. Ini memungkinkan Anda untuk mengganti rahasia jangka panjang dengan rahasia jangka pendek, secara signifikan mengurangi risiko kompromi. Karena kredenSIAL tidak lagi disimpan dengan aplikasi, merotasi kredenSIAL tidak lagi memerlukan pembaruan aplikasi Anda dan menerapkan perubahan ke klien aplikasi.

Untuk jenis rahasia lain yang mungkin Anda miliki di organisasi Anda:

- AWS kredenSIAL - Kami merekomendasikan AWS Identity and Access Management.
- Kunci enkripsi - Kami sarankan AWS Key Management Service.
- Kunci SSH — Kami merekomendasikan Amazon EC2 Instance Connect.
- Kunci dan sertifikat pribadi - Kami merekomendasikan AWS Certificate Manager.

Memulai dengan Secrets Manager

Jika Anda baru mengenal Secrets Manager, mulailah dengan salah satu tutorial berikut:

- the section called “Ganti rahasia hardcode ”
- the section called “Ganti kredensi DB hardcode ”
- the section called “Rotasi pengguna bergantian”
- the section called “Rotasi pengguna tunggal”

Tugas lain yang dapat Anda lakukan dengan rahasia:

- Kelola rahasia
- Kontrol akses ke rahasia Anda
- Dapatkan rahasia
- Merotasi rahasia
- Pemberitahuan acara rahasia
- Memantau rahasia
- Pantau rahasia untuk kepatuhan
- Buat rahasia di AWS CloudFormation

Kepatuhan dengan standar

AWS Secrets Manager telah menjalani audit untuk berbagai standar dan dapat menjadi bagian dari solusi Anda ketika Anda perlu mendapatkan sertifikasi kepatuhan. Untuk informasi selengkapnya, lihat Validasi kepatuhan.

Harga

Saat Anda menggunakan Secrets Manager, Anda hanya membayar untuk apa yang Anda gunakan, tanpa biaya minimum atau pengaturan. Tidak ada biaya untuk rahasia yang ditandai untuk dihapus. Untuk daftar harga lengkap saat ini, lihat AWS Secrets Manager Harga. Untuk memantau biaya Anda, lihatthe section called “Biaya Monitor Secrets Manager”.

Anda dapat menggunakan Kunci yang dikelola AWS `aws/secretsmanager` yang dibuat oleh Secrets Manager untuk mengenkripsi rahasia Anda secara gratis. Jika Anda membuat kunci KMS Anda sendiri untuk mengenkripsi rahasia Anda, Anda akan AWS dikenakan biaya dengan AWS KMS tarif saat ini. Untuk informasi selengkapnya, silakan lihat AWS Key Management Service Harga.

Saat Anda mengaktifkan rotasi otomatis (kecuali rotasi terkelola), Secrets Manager menggunakan AWS Lambda fungsi untuk memutar rahasia, dan Anda akan dikenakan biaya untuk fungsi rotasi pada tingkat Lambda saat ini. Untuk informasi selengkapnya, silakan lihat AWS Lambda Harga.

Jika Anda mengaktifkan AWS CloudTrail di akun Anda, Anda dapat memperoleh log panggilan API yang dikirim oleh Secrets Manager. Manajer Rahasia mencatat semua peristiwa sebagai peristiwa manajemen. AWS CloudTrail menyimpan salinan pertama dari semua acara manajemen secara gratis. Namun, Anda dapat dikenakan biaya untuk Amazon S3 untuk penyimpanan log dan untuk Amazon SNS jika Anda mengaktifkan notifikasi. Juga, jika Anda mengatur jalur tambahan, salinan

tambahan dari acara manajemen dapat menimbulkan biaya. Untuk informasi selengkapnya, lihat harga AWS CloudTrail.

Anda dapat menggunakan tag alokasi biaya di Manajer Rahasia untuk melacak dan mengkategorikan pengeluaran yang terkait dengan rahasia atau proyek tertentu. Untuk informasi selengkapnya, lihat the section called “Kata kunci rahasia” panduan ini dan Menggunakan tag alokasi AWS biaya di Panduan AWS Billing Pengguna.

Akses AWS Secrets Manager

Anda dapat bekerja dengan Manajer Rahasia dengan salah satu cara berikut:

- Konsol Manajer Rahasia
- Alat baris perintah
- AWS SDK
- API Kueri HTTPS
- AWS Secrets Manager titik akhir

Konsol Manajer Rahasia

Anda dapat mengelola rahasia Anda menggunakan konsol Manajer Rahasia berbasis browser dan melakukan hampir semua tugas yang terkait dengan rahasia Anda dengan menggunakan konsol.

Alat baris perintah

Alat AWS baris perintah memungkinkan Anda mengeluarkan perintah di baris perintah sistem Anda untuk melakukan Manajer Rahasia dan AWS tugas-tugas lainnya. Ini mungkin lebih cepat dan nyaman dibandingkan jika menggunakan konsol. Alat baris perintah dapat berguna jika Anda ingin membuat skrip untuk melakukan AWS tugas.

Ketika Anda memasukkan perintah dalam shell perintah, ada risiko riwayat perintah diakses atau utilitas memiliki akses ke parameter perintah Anda. Lihat the section called “Mengurangi risiko menggunakan AWS CLI untuk menyimpan rahasia Anda AWS Secrets Manager”.

Alat baris perintah secara otomatis menggunakan titik akhir default untuk layanan di Wil AWS ayah. Anda dapat menentukan titik akhir yang berbeda untuk permintaan API Anda. Lihat the section called “Titik akhir Manajer Rahasia”.

AWS menyediakan dua set alat baris perintah:

- AWS Command Line Interface (AWS CLI)
- AWS Tools for Windows PowerShell

AWS SDK

AWS SDK terdiri dari pustaka dan kode sampel untuk berbagai bahasa pemrograman dan platform. SDK mencakup tugas-tugas seperti permintaan penandatanganan kriptografis, mengelola kesalahan, dan mencoba kembali permintaan secara otomatis. Untuk mengunduh dan menginstal salah satu SDK, lihat Alat untuk Amazon Web Services.

AWS SDK secara otomatis menggunakan titik akhir default untuk layanan di Wil AWS ayah. Anda dapat menentukan titik akhir yang berbeda untuk permintaan API Anda. Lihat the section called “Titik akhir Manajer Rahasia”.

Untuk dokumentasi SDK, lihat:

- C++
- Go
- Java
- JavaScript
- Kotlin
- .NET
- PHP
- Python (Voto3)
- Ruby
- Karat
- SAP ABAP
- Cepat

API Kueri HTTPS

HTTPS Query API memberi Anda akses terprogram ke Secrets Manager dan AWS. HTTPS Query API memungkinkan Anda untuk mengeluarkan permintaan HTTPS langsung ke layanan.

Meskipun Anda dapat melakukan panggilan langsung ke Secrets Manager HTTPS Query API, sebaiknya gunakan salah satu SDK sebagai gantinya. SDK melakukan banyak tugas berguna yang harus Anda lakukan secara manual. Misalnya, SDK secara otomatis menandatangani permintaan Anda dan mengubah tanggapan menjadi struktur yang secara sintaksis sesuai dengan bahasa Anda.

Untuk melakukan panggilan HTTPS ke Manajer Rahasia, Anda terhubung ke???.

AWS Secrets Manager titik akhir

Untuk terhubung secara terprogram ke Secrets Manager, Anda menggunakan titik akhir, URL titik masuk untuk layanan. Titik akhir Secrets Manager adalah titik akhir dual-stack, yang berarti mereka mendukung IPv4 dan IPv6.

Secrets Manager menawarkan titik akhir yang mendukung Federal Information Processing Standard (FIPS) 140-2 di beberapa Wilayah.

Secrets Manager mendukung TLS 1.2 dan 1.3. Secrets Manager mendukung PQTLS di semua wilayah kecuali Wilayah China.

> **Note**
>
> Python AWS SDK dan AWS CLI upaya untuk memanggil IPv6 dan kemudian IPv4 secara berurutan, jadi jika Anda tidak mengaktifkan IPv6, mungkin perlu beberapa waktu sebelum waktu panggilan habis dan mencoba lagi dengan IPv4. Untuk mengatasi masalah ini, Anda dapat menonaktifkan IPv6 sepenuhnya atau bermigrasi ke IP v6.

Berikut ini adalah titik akhir layanan untuk Manajer Rahasia. Perhatikan bahwa penamaan berbeda dari konven si penamaan dual-stack yang khas. Untuk informasi tentang menggunakan pengalamatan tumpukan ganda di Secrets Manager, lihatAkses IPv4 dan IPv6.

Nama Wilayah	Wilayah	Titik Akhir	Protokol	
AS Timur (Ohio)	us–east-2	secretsmanager.us-east-2.amazonaws.com secretsmanager-fips.us-east-2.amazonaws.com	HTTPS HTTPS	
AS Timur (Virginia Utara)	us-east-1	secretsmanager.us-east-1.amazonaws.com secretsmanager-fips.us-east-1.amazonaws.com	HTTPS HTTPS	

Nama Wilayah	Wilayah	Titik Akhir	Protokol
AS Barat (California Utara)	us-west-1	secretsmanager.us-west-1.amazonaws.com secretsmanager-fips.us-west-1.amazonaws.com	HTTPS HTTPS
US West (Oregon)	us-west-2	secretsmanager.us-west-2.amazonaws.com secretsmanager-fips.us-west-2.amazonaws.com	HTTPS HTTPS
Africa (Cape Town)	af-south-1	secretsmanager.af-south-1.amazonaws.com	HTTPS
Asia Pasifik (Hong Kong)	ap-east-1	secretsmanager.ap-east-1.amazonaws.com	HTTPS
Asia Pasifik (Hyderabad)	ap-south-2	secretsmanager.ap-south-2.amazonaws.com	HTTPS
Asia Pasifik (Jakarta)	ap-southeast-3	secretsmanager.ap-southeast-3.amazonaws.com	HTTPS
Asia Pasifik (Malaysia)	ap-southeast-5	secretsmanager.ap-southeast-5.amazonaws.com	HTTPS

Nama Wilayah	Wilayah	Titik Akhir	Protokol	
Asia Pasifik (Melbourne)	ap-southeast-4	secretsmanager.ap-southeast-4.amazonaws.com	HTTPS	
Asia Pasifik (Mumbai)	ap-south-1	secretsmanager.ap-south-1.amazonaws.com	HTTPS	
Asia Pasifik (Selandia Baru)	ap-Southeast-6	secretsmanager.ap-southeast-6.amazonaws.com	HTTPS	
Asia Pasifik (Osaka)	ap-northeast-3	secretsmanager.ap-northeast-3.amazonaws.com	HTTPS	
Asia Pasifik (Seoul)	ap-northeast-2	secretsmanager.ap-northeast-2.amazonaws.com	HTTPS	
Asia Pasifik (Singapura)	ap-southeast-1	secretsmanager.ap-southeast-1.amazonaws.com	HTTPS	
Asia Pacific (Sydney)	ap-southeast-2	secretsmanager.ap-southeast-2.amazonaws.com	HTTPS	
Asia Pasifik (Taipei)	ap-east-2	secretsmanager.ap-east-2.amazonaws.com	HTTPS	

Nama Wilayah	Wilayah	Titik Akhir	Protokol
Asia Pasifik (Thailand)	ap-Southeast-7	secretsmanager.ap-southeast-7.amazonaws.com	HTTPS
Asia Pacific (Tokyo)	ap-northeast-1	secretsmanager.ap-northeast-1.amazonaws.com	HTTPS
Canada (Central)	ca-central-1	secretsmanager.ca-central-1.amazonaws.com secretsmanager-fips.ca-central-1.amazonaws.com	HTTPS HTTPS
Kanada Barat (Calgary)	ca-west-1	secretsmanager.ca-west-1.amazonaws.com secretsmanager-fips.ca-west-1.amazonaws.com	HTTPS HTTPS
Eropa (Frankfurt)	eu-central-1	secretsmanager.eu-central-1.amazonaws.com	HTTPS
Eropa (Irlandia)	eu-west-1	secretsmanager.eu-west-1.amazonaws.com	HTTPS
Europe (London)	eu-west-2	secretsmanager.eu-west-2.amazonaws.com	HTTPS
Eropa (Milan)	eu-south-1	secretsmanager.eu-south-1.amazonaws.com	HTTPS
Eropa (Paris)	eu-west-3	secretsmanager.eu-west-3.amazonaws.com	HTTPS

Nama Wilayah	Wilayah	Titik Akhir	Protokol
Eropa (Spanyol)	eu-south-2	secretsmanager.eu-south-2.amazonaws.com	HTTPS
Eropa (Stockholm)	eu-north-1	secretsmanager.eu-north-1.amazonaws.com	HTTPS
Eropa (Zürich)	eu-central-2	secretsmanager.eu-central-2.amazonaws.com	HTTPS
Israel (Tel Aviv)	il-central-1	secretsmanager.il-central-1.amazonaws.com	HTTPS
Meksiko (Tengah)	mx-sentral-1	secretsmanager.mx-central-1.amazonaws.com	HTTPS
Timur Tengah (Bahrain)	me-south-1	secretsmanager.me-south-1.amazonaws.com	HTTPS
Timur Tengah (UAE)	me-central-1	secretsmanager.me-central-1.amazonaws.com	HTTPS
Amerika Selatan (Sao Paulo)	sa-east-1	secretsmanager.sa-east-1.amazonaws.com	HTTPS
AWS GovCloud (US-East)	us-gov-east-1	secretsmanager.us-gov-east-1.amazonaws.com secretsmanager-fips.us-gov-east-1.amazonaws.com	HTTPS HTTPS

Nama Wilayah	Wilayah	Titik Akhir	Protokol	
AWS GovCloud (US-West)	us-gov-west-1	secretsmanager.us-gov-west-1.amazonaws.com secretsmanager-fips.us-gov-west-1.amazonaws.com	HTTPS HTTPS	

AWS Secrets Manager praktik terbaik

Secrets Manager menyediakan sejumlah fitur keamanan untuk dipertimbangkan saat Anda mengembangkan dan menerapkan kebijakan keamanan Anda sendiri. Praktik terbaik berikut adalah pedoman umum dan tidak mewakili solusi keamanan yang lengkap. Karena praktik terbaik ini mungkin tidak sesuai atau cukup untuk lingkungan Anda, anggap sebagai pertimbangan yang membantu dan bukan sebagai resep.

Pertimbangkan praktik terbaik berikut untuk menyimpan dan mengelola rahasia:

- Menyimpan kredensi dan informasi sensitif lainnya di AWS Secrets Manager
- Temukan rahasia yang tidak dilindungi dalam kode Anda
- Pilih kunci enkripsi untuk rahasia Anda
- Gunakan caching untuk mengambil rahasia
- Putar rahasia Anda
- Mengurangi risiko menggunakan CLI
- Batasi akses ke rahasia
- Replikasi rahasia
- Memantau rahasia
- Jalankan infrastruktur Anda di jaringan pribadi

Menyimpan kredensi dan informasi sensitif lainnya di AWS Secrets Manager

Secrets Manager dapat membantu meningkatkan postur keamanan dan kepatuhan Anda, serta mengurangi risiko akses tidak sah ke informasi sensitif Anda. Secrets Manager mengenkripsi rahasia saat istirahat menggunakan kunci enkripsi yang Anda miliki dan simpan di AWS Key Management Service ()AWS KMS. Saat Anda mengambil rahasia, Secrets Manager mendekripsi rahasia dan mengirimkannya dengan aman melalui TLS ke lingkungan lokal Anda. Untuk informasi selengkapnya, lihat Buat rahasia.

Temukan rahasia yang tidak dilindungi dalam kode Anda

CodeGuru Reviewer terintegrasi dengan Secrets Manager untuk menggunakan detektor rahasia yang menemukan rahasia yang tidak dilindungi dalam kode Anda. Detektor rahasia mencari kata sandi hardcode, string koneksi database, nama pengguna, dan banyak lagi. Untuk informasi selengkapnya, lihat the section called “ CodeGuru Peninjau Amazon”.

Amazon Q dapat memindai basis kode Anda untuk mencari kerentanan keamanan dan masalah kualitas kode untuk meningkatkan postur aplikasi Anda selama siklus pengembangan. Untuk informasi selengkapnya, lihat Memindai kode Anda dengan Amazon Q di Panduan Pengguna Pengembang Amazon Q.

Pilih kunci enkripsi untuk rahasia Anda

Untuk kebanyakan kasus, kami sarankan menggunakan kunci `aws/secretsmanager` AWS terkelola untuk mengenkripsi rahasia. Tidak ada biaya untuk menggunakannya.

Untuk dapat mengakses rahasia dari akun lain atau menerapkan kebijakan kunci ke kunci enkripsi, gunakan kunci yang dikelola pelanggan untuk mengenkripsi rahasia.

- Dalam kebijakan kunci, tetapkan nilai `secretsmanager.<region>.amazonaws.com` ke kunci `kms:ViaService`kondisi. Ini membatasi penggunaan kunci hanya untuk permintaan dari Secrets Manager.
- Untuk lebih membatasi penggunaan kunci hanya permintaan dari Secrets Manager dengan konteks yang benar, gunakan kunci atau nilai dalam konteks enkripsi Secrets Manager sebagai syarat untuk menggunakan kunci KMS dengan membuat:
  - Operator kondisi string dalam kebijakan IAM atau kebijakan kunci
  - Kendala hibah dalam hibah

Untuk informasi selengkapnya, lihat the section called “Enkripsi rahasia dan dekripsi”.

Gunakan caching untuk mengambil rahasia

Untuk menggunakan rahasia Anda dengan paling efisien, kami sarankan Anda menggunakan salah satu komponen caching Secrets Manager yang didukung berikut ini untuk menyimpan rahasia Anda dan memperbaruinya hanya jika diperlukan:

- Java dengan caching sisi klien
- Python dengan caching sisi klien
- .NET dengan caching sisi klien
- Pergi dengan caching sisi klien
- Karat dengan caching sisi klien
- AWS Parameter dan Rahasia Ekstensi Lambda
- the section called “Amazon EKS”
- Gunakan the section called “AWS Penyedia Kredensia Beban Kerja” untuk membakukan konsumsi rahasia dari Secrets Manager di seluruh lingkungan seperti AWS Lambda Amazon Elastic Container Service, Amazon Elastic Kubernetes Service, dan Amazon Elastic Compute Cloud.

Putar rahasia Anda

Jika Anda tidak mengubah rahasia Anda untuk jangka waktu yang lama, rahasia menjadi lebih mungkin untuk dikompromikan. Dengan Secrets Manager, Anda dapat mengatur rotasi otomatis sesering setiap empat jam. Secrets Manager menawarkan dua strategi untuk rotasi: Pengguna tunggal danPengguna bergantian. Untuk informasi selengkapnya, lihat Merotasi rahasia .

Mengurangi risiko menggunakan CLI

Saat Anda menggunakan AWS operasi AWS CLI untuk memanggil, Anda memasukkan perintah tersebut di shell perintah. Sebagian besar shell perintah menawarkan fitur yang dapat membahayakan rahasia Anda, seperti logging dan kemampuan untuk melihat perintah yang terakhir dimasukkan. Sebelum Anda menggunakan AWS CLI untuk memasukkan informasi sensitif, pastikan untukthe section called “Mengurangi risiko menggunakan AWS CLI untuk menyimpan rahasia Anda AWS Secrets Manager”.

Batasi akses ke rahasia

Dalam pernyataan kebijakan IAM yang mengontrol akses ke rahasia Anda, gunakan prinsip akses yang paling tidak memiliki hak istimewa. Anda dapat menggunakan peran dan kebijakan IAM, kebijakan sumber daya, dan kontrol akses berbasis atribut (ABAC). Untuk informasi selengkapnya, lihat the section called “Kontrol autentikasi dan akses”.

Topik

- Blokir akses luas ke rahasia
- Berhati-hatilah dengan kondisi alamat IP dalam kebijakan
- Batasi permintaan dengan kondisi titik akhir VPC

Blokir akses luas ke rahasia

Dalam kebijakan identitas yang memungkinkan tindakan`PutResourcePolicy`, kami sarankan Anda menggunakannya`BlockPublicPolicy: true`. Kondisi ini berarti bahwa pengguna hanya dapat melampirkan kebijakan sumber daya ke rahasia jika kebijakan tidak mengizinkan akses luas.

Secrets Manager menggunakan penalaran otomatis Zelkova untuk menganalisis kebijakan sumber daya untuk akses luas. Untuk informasi selengkapnya tentang Zelkova, lihat Cara AWS menggunakan penalaran otomatis untuk membantu Anda mencapai keamanan dalam skala besar di Blog Keamanan. AWS

Contoh berikut menunjukkan cara menggunakan`BlockPublicPolicy`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "secretsmanager:PutResourcePolicy",
        "Resource": "arn:aws:secretsmanager:us-
east-1:123456789012:secret:secretName-AbCdEf",
    "Condition": {
            "Bool": {
                "secretsmanager:BlockPublicPolicy": "true"
            }
        }
    }
}
```

Berhati-hatilah dengan kondisi alamat IP dalam kebijakan

Berhati-hatilah saat Anda menentukan operator kondisi alamat IP atau kunci `aws:SourceIp` kondisi dalam pernyataan kebijakan yang mengizinkan atau menolak akses ke Secrets Manager.

Misalnya, jika Anda melampirkan kebijakan yang membatasi AWS tindakan untuk permintaan dari rentang alamat IP jaringan perusahaan Anda ke rahasia, maka permintaan Anda sebagai pengguna IAM yang menjalankan permintaan dari jaringan perusahaan berfungsi seperti yang diharapkan. Namun, jika Anda mengaktifkan layanan lain untuk mengakses rahasia atas nama Anda, seperti saat Anda mengaktifkan rotasi dengan fungsi Lambda, fungsi tersebut akan memanggil operasi Secrets Manager dari ruang alamat AWS-internal. Permintaan yang terkena dampak kebijakan dengan filter alamat IP gagal.

Selain itu, kunci `aws:sourceIP` kondisi kurang efektif ketika permintaan berasal dari titik akhir VPC Amazon. Untuk membatasi permintaan ke titik akhir VPC tertentu, gunakan. the section called “Batasi permintaan dengan kondisi titik akhir VPC”

Batasi permintaan dengan kondisi titik akhir VPC

Untuk mengizinkan atau menolak akses ke permintaan dari titik akhir VPC atau VPC tertentu, gunakan `aws:SourceVpc` untuk membatasi akses ke permintaan dari VPC yang ditentukan atau `aws:SourceVpce` untuk membatasi akses ke permintaan dari titik akhir VPC yang ditentukan. Lihat the section called “Contoh: Izin dan VPC”.

- `aws:SourceVpc` membatasi akses ke permintaan dari VPC yang ditentukan.
- `aws:SourceVpce` membatasi akses ke permintaan dari VPC endpoint yang ditentukan.

Jika Anda menggunakan kunci kondisi ini dalam pernyataan kebijakan sumber daya yang mengizinkan atau menolak akses ke rahasia Secrets Manager, Anda dapat secara tidak sengaja menolak akses ke layanan yang menggunakan Secrets Manager untuk mengakses rahasia atas nama Anda. Hanya beberapa AWS layanan yang dapat berjalan dengan titik akhir dalam VPC Anda. Jika Anda membatasi permintaan rahasia ke titik akhir VPC atau VPC, maka panggilan ke Secrets Manager dari layanan yang tidak dikonfigurasi untuk layanan dapat gagal.

Lihat the section called “Titik akhir VPC (AWS PrivateLink)”.

Replikasi rahasia

Secrets Manager dapat secara otomatis mereplikasi rahasia Anda ke beberapa AWS Wilayah untuk memenuhi persyaratan ketahanan atau pemulihan bencana Anda. Untuk informasi selengkapnya, lihat Multi-region replikasi.

Memantau rahasia

Secrets Manager memungkinkan Anda untuk mengaudit dan memantau rahasia melalui integrasi dengan layanan AWS pencatatan, pemantauan, dan pemberitahuan. Untuk informasi lebih lanjut, lihat:

- the section called “Masuk dengan AWS CloudTrail ”
- the section called “Monitor dengan CloudWatch”
- the section called “Pantau rahasia untuk kepatuhan”
- the section called “Biaya Monitor Secrets Manager”
- the section called “Mendeteksi ancaman dengan GuardDuty”

Jalankan infrastruktur Anda di jaringan pribadi

Kami menyarankan Anda menjalankan infrastruktur sebanyak mungkin di jaringan pribadi yang tidak dapat diakses dari internet publik. Anda dapat membuat koneksi pribadi antara VPC dan Secrets Manager dengan membuat antarmuka VPC endpoint. Lihat informasi yang lebih lengkap di the section called “Titik akhir VPC (AWS PrivateLink)”.

Kelola kunci API untuk beban kerja yang sensitif terhadap keamanan

AWS Secrets Manager membantu Anda mengelola, mengambil, dan memutar kredenSIAL database, kredenSIAL aplikasi, token OAuth, kunci API, dan rahasia lainnya sepanjang siklus hidupnya. Halaman ini memberikan panduan preskriptif untuk mengelola kunci API dan kredenSIAL pihak ketiga dalam beban kerja yang sensitif terhadap keamanan. Gabungkan rotasi otomatis dengan AWS KMS kunci yang dikelola pelanggan dan kebijakan IAM dengan hak istimewa terendah. Konfigurasikan juga akses titik akhir VPC untuk meminimalkan jendela eksposur dan radius ledakan dari kredensia yang dikompromikan.

Simpan kunci API di AWS Secrets Manager

Bagian ini menjelaskan cara menyusun rahasia kunci API Anda untuk rotasi dan pengambilan yang optimal. Simpan setiap kunci API sebagai rahasia terpisah dengan nilai JSON terstruktur. Dengan struktur ini, aplikasi Anda dapat mengambil bidang individual, dan Manajer Rahasia dapat meneruskan nilai yang benar ke fungsi rotasi.

Contoh berikut menunjukkan struktur JSON umum untuk rahasia kunci API pihak ketiga. Bidang aktual bergantung pada persyaratan penyedia Anda — tinjau dokumentasi penyedia untuk kredenSIAL dan metadata spesifik yang perlu Anda simpan.

```
{
  "apiKey": "your-api-key-value",
  "apiKeyId": "key-identifier",
  "endpoint": "https://api.example.com/v1",
  "provider": "example-service"
}
```

Contoh struktur rahasia untuk kunci API

Bidang	Nilai contoh	Tujuan
`apiKey`	`sk_live_abc123...`	Nilai kredensi yang digunakan aplikasi Anda untuk mengotentikasi dengan API pihak ketiga.

Bidang	Nilai contoh	Tujuan
`apiKeyId`	`key_001`	Pengidentifikasi untuk kunci di sisi penyedia. Digunakan selama rotasi untuk membuat kunci baru dan menghapus yang lama.
`endpoint`	`https://api.example.com/v1`	URL titik akhir API. Simpan dengan kunci sehingga pengambilan mengembalikan semua yang dibutuhkan aplikasi untuk terhubung.
`provider`	`stripe`	Nama penyedia. Berguna untuk menandai, memfilter, dan logika fungsi rotasi yang menangani beberapa penyedia.

Tandai setiap rahasia dengan metadata untuk mendukung kondisi kebijakan IAM dan penyaringan organisasi. Misalnya, Anda dapat menandai rahasia dengan tim pemilik, lingkungan, dan cakupan kepatuhan menggunakan konsol AWS CLI or:

```
aws secretsmanager tag-resource \
    --secret-id prod/payments/stripe-api-key \
    --tags Key=Team,Value=payments Key=Environment,Value=production \
           Key=Provider,Value=stripe Key=Compliance,Value=pci-dss
```

Untuk informasi lebih lanjut tentang membuat rahasia, lihatBuat rahasia.

Konfigurasi enkripsi

Secrets Manager mengenkripsi setiap nilai rahasia saat diam menggunakan AWS KMS kunci. Untuk beban kerja yang sensitif terhadap keamanan, pilih kunci enkripsi berdasarkan persyaratan kepatuhan dan kontrol akses Anda.

Opsi kunci KMS untuk rahasia kunci API

Tipe Kunci	Kapan harus digunakan	Pertimbangan keamanan
AWS kunci terkelola	Default untuk sebagian besar beban kerja. Tidak ada biaya	Kebijakan kunci dibatasi hanya untuk operasi Manajer Rahasia dan tidak dapat

Tipe Kunci	Kapan harus digunakan	Pertimbangan keamanan
(`aws/secretsmanager`)	tambahan atau overhead manajemen kunci.	dimodifikasi. Tidak dapat digunakan untuk akses lintas akun.
Kunci yang dikelola pelanggan	Persyaratan kepatuhan (misalnya, PCI DSS, HIPAA, SOC 2, atau standar lain yang berlaku). Cross-account berbagi rahasia. Persyaratan audit penggunaan utama.	Anda mengontrol kebijakan utama. Anda dapat membatasi prinsipal mana yang dapat mendekripsi. Anda dapat menonaktifkan atau menjadwalkan penghapusan kunci secara independen dari rahasia. Menyediakan jejak audit independen melalui.

Untuk beban kerja yang sensitif terhadap keamanan, gunakan kunci yang dikelola pelanggan dengan kondisi kebijakan utama berikut:

- `kms:ViaService`— Batasi penggunaan kunci untuk permintaan yang berasal dari Secrets Manager (`secretsmanager.<region>.amazonaws.com`).
- `kms:EncryptionContext:SecretARN`— Batasi dekripsi ke ARN rahasia tertentu dengan mencocokkan konteks enkripsi Manajer Rahasia.
- Kunci terpisah per batas kepatuhan — Gunakan AWS KMS kunci yang berbeda untuk rahasia dalam lingkup kepatuhan yang berbeda (misalnya, PCI dibandingkan dengan non-PCI).

Untuk penjelasan lengkap tentang proses enkripsi dan dekripsi, lihatthe section called “Enkripsi rahasia dan dekripsi”.

Rotasi otomatis untuk kunci API

Rotasi otomatis mengurangi jendela eksposur kredenSIAL yang dikompromikan. Secrets Manager memanggil fungsi Lambda pada jadwal. Fungsi ini membuat kunci API baru di penyedia, memperbarui nilai rahasia, dan menghapus kunci lama.

Secrets Manager menyediakan fungsi rotasi terkelola untuk beberapa penyedia pihak ketiga melalui rahasia eksternal yang dikelola. Untuk informasi selengkapnya tentang fitur ini dan daftar penyedia yang didukung, lihatthe section called “Mitra Integrasi”. Untuk penyedia tanpa dukungan rotasi terkelola, Anda menerapkan fungsi rotasi Lambda khusus yang memanggil API penyedia untuk membuat dan menghapus kunci.

Siklus hidup fungsi rotasi

Fungsi rotasi Lambda mengimplementasikan empat langkah. Secrets Manager memanggil fungsi sekali untuk setiap langkah, melewati parameter. `Step` Jika ada langkah yang gagal, Manajer Rahasia secara otomatis mencoba ulang seluruh rotasi.

Langkah-langkah rotasi untuk kunci API

Langkah	Tindakan untuk kunci API	Penanganan kegagalan
`createSecret`	Panggil API penyedia untuk membuat kunci baru. Simpan nilai kunci baru di Secrets Manager dengan label p `AWSPENDING` ementasan.	Jika pembuatan kunci gagal, rotasi tidak beralih ke langkah berikutnya. Kunci yang ada tetap aktif sebagai`AWSCURRENT` .
`setSecret`	Untuk kunci API yang dibuat di penyedia, langkah ini biasanya tidak dilakukan. Langkah ini digunakan ketika kunci acak dihasilkan di Secrets Manager dan perlu diatur di penyedia — yang bukan aliran kunci API yang khas.	Jika langkah ini gagal, rotasi tidak dilanjutkan ke`testSecret` .
`testSecret`	Ambil `AWSPENDING` nilai dari Secrets Manager dan buat panggilan API uji ke penyedia untuk memverifikasi kunci baru berfungsi.	Jika pengujian gagal, hapus kunci tertunda di penyedia dan angkat pengecualian.
`finishSecret`	P `AWSCURRENT` indah ke kunci baru. Kunci lama bergerak ke`AWSPREVIOUS` . Secara opsional menghapus kunci lama di penyedia.	Jika pembaruan label gagal, rotasi tidak selesai. Kunci baru ada tetapi belum diberi label`AWSCURRENT` .

Untuk template fungsi rotasi lengkap dan panduan implementasi, lihatthe section called “Fungsi rotasi Lambda”.

Konfigurasi jadwal rotasi

Tetapkan interval rotasi berdasarkan persyaratan kepatuhan dan kebijakan keamanan internal Anda. Lihat standar kepatuhan yang berlaku untuk beban kerja Anda untuk menentukan frekuensi rotasi yang sesuai.

Gunakan jendela rotasi untuk mengontrol kapan rotasi terjadi. Ini mencegah rotasi berjalan selama puncak lalu lintas atau jendela pemeliharaan:

```
aws secretsmanager rotate-secret \
    --secret-id prod/payments/stripe-api-key \
    --rotation-rules '{
        "ScheduleExpression": "cron(0 4 ? * SUN *)",
        "Duration": "2h"
    }'
```

Untuk sintaks ekspresi jadwal, lihatthe section called “Jadwal rotasi”.

Mengambil rahasia secara efisien

Secrets Manager mendukung 10.000 transaksi per detik pada `GetSecretValue` panggilan. Sebagian besar aplikasi tidak mengalami pelambatan. Untuk aplikasi dengan volume panggilan yang sangat tinggi atau jalur sensitif latensi, gunakan solusi caching untuk mengurangi panggilan API dan meningkatkan waktu respons.

Secrets Manager menyediakan klien caching untuk beberapa bahasa, serta ekstensi Lambda yang menyimpan rahasia secara lokal dalam lingkungan eksekusi. Untuk informasi selengkapnya tentang opsi caching, lihatthe section called “Java dengan caching sisi klien”,the section called “Python dengan caching sisi klien”, danthe section called “Pergi dengan caching sisi klien”.

Untuk semua pola akses, konfigurasikan kebijakan IAM yang `secretsmanager:GetSecretValue` membatasi rahasia spesifik yang dibutuhkan setiap aplikasi:

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": "secretsmanager:GetSecretValue",
    "Resource": "arn:aws:secretsmanager:us-east-1:123456789012:secret:prod/payments/*",
```

```
    "Condition": {
      "StringEquals": {
        "aws:ResourceTag/Environment": "production"
      }
    }
  }]
}
```

Pengerasan keamanan untuk beban kerja sensitif

Praktik berikut memberikan pertahanan mendalam untuk kunci API di lingkungan dengan persyaratan keamanan yang ketat (misalnya, PCI DSS, SOC 2, HIPAA, atau kerangka kerja kepatuhan lain yang berlaku):

Batasi akses jaringan dengan titik akhir VPC

Buat titik akhir VPC antarmuka untuk Manajer Rahasia sehingga pengambilan rahasia tidak pernah melintasi internet publik. Terapkan kebijakan titik akhir yang membatasi rahasia mana yang dapat diakses melalui titik akhir. Untuk informasi selengkapnya, lihat the section called “Titik akhir VPC (AWS PrivateLink)”.

Menerapkan kebijakan sumber daya ke rahasia

Lampirkan kebijakan sumber daya ke setiap rahasia yang secara eksplisit menolak akses dari kepala sekolah di luar akun Anda atau di luar titik akhir VPC tertentu. Ini memberikan batas otorisasi kedua yang independen dari kebijakan identitas IAM.

Memantau akses rahasia dengan

secara otomatis mencatat semua panggilan API Secrets Manager, termasuk`GetSecretValue`,`PutSecretValue`, dan`RotateSecret`. Untuk rahasia yang sensitif terhadap keamanan, buat CloudWatch alarm Amazon yang memicu `GetSecretValue` panggilan tak terduga — misalnya, panggilan dari alamat IP sumber yang tidak dikenal atau prinsipal IAM.

Gunakan **`AWSPREVIOUS`** untuk rotasi anggun

Selama rotasi, Secrets Manager mempertahankan nilai kunci sebelumnya dengan label p `AWSPREVIOUS` ementasan. Jika penyedia Anda membatalkan kunci sebelumnya saat kunci baru dibuat, konfigurasikan aplikasi Anda untuk kembali ke `AWSPREVIOUS` jika `AWSCURRENT` mengembalikan kesalahan otentikasi. Ini mencegah downtime selama jendela singkat antara pembuatan kunci dan pembaruan label.

Validasi kebijakan sumber daya sebelum melampirkan

Rahasia tanpa kebijakan sumber daya sudah memblokir akses publik. Saat Anda melampirkan kebijakan sumber daya ke rahasia Anda, gunakan `ValidateResourcePolicy` API untuk memastikan kebijakan Anda tidak memberikan akses publik yang luas. Anda juga dapat menggunakan `BlockPublicPolicy` parameter with `PutResourcePolicy` untuk mencegah melampirkan kebijakan yang memberikan akses publik. Gunakan kunci `aws:PrincipalOrgID` kondisi dalam kebijakan sumber daya untuk mencegah akses dari kepala sekolah di luar organisasi Anda.

Pertanyaan umum

Bagian ini menjawab pertanyaan umum tentang rotasi kunci API dan manajemen rahasia di AWS Secrets Manager.

Bagaimana cara menangani periode tumpang tindih selama rotasi?

Ini hanya diperlukan jika penyedia membatalkan kunci yang ada saat kunci baru dibuat. Jika penyedia mendukung beberapa kunci aktif secara bersamaan, kedua kunci bekerja selama periode rotasi tanpa logika fallback sisi aplikasi. Untuk penyedia yang membatalkan kunci lama, konfigurasikan aplikasi Anda untuk mencoba lagi `AWSPREVIOUS` jika kunci saat ini mengembalikan kesalahan 401 atau 403. Hapus kunci lama di penyedia pada `finishSecret` langkah hanya setelah Anda mengonfirmasi kunci baru berfungsi.

Bagaimana jika penyedia saya tidak mendukung pembuatan kunci terprogram?

Jika penyedia memerlukan pembuatan kunci manual (melalui konsol web, misalnya), Anda tidak dapat sepenuhnya mengotomatiskan rotasi. Sebagai gantinya, gunakan fungsi rotasi yang mengirimkan pemberitahuan (melalui Amazon Simple Notification Service) saat rotasi jatuh tempo, meminta operator untuk membuat kunci secara manual dan memperbarui nilai rahasia. Tetapkan jadwal rotasi agar sesuai dengan persyaratan rotasi kepatuhan Anda dan gunakan CloudWatch alarm Amazon aktif `days_since_last_rotation` untuk mendeteksi rotasi yang terlewat.

Bagaimana cara menghindari pelambatan API saat mengambil rahasia?

Secrets Manager mendukung 10.000 transaksi per detik`GetSecretValue`. Sebagian besar aplikasi tidak mengalami pelambatan. Jika aplikasi Anda membuat volume panggilan yang sangat tinggi,

gunakan klien caching atau ekstensi Parameter dan Rahasia Lambda. Ini menyimpan nilai rahasia dalam memori dan menyegarkan secara berkala, mengurangi jumlah panggilan API. Atur cache TTL ke nilai yang lebih pendek dari interval rotasi Anda sehingga aplikasi mengambil kunci baru setelah rotasi.

Haruskah saya menggunakan satu rahasia per lingkungan atau satu rahasia dengan versi?

Gunakan rahasia terpisah untuk setiap lingkungan (misalnya, `prod/payments/stripe` dan`dev/payments/stripe`). Hal ini memungkinkan kebijakan IAM yang berbeda, jadwal rotasi, dan kunci enkripsi per lingkungan. Versi rahasia (label pementasan) adalah untuk manajemen status rotasi, bukan pemisahan lingkungan.

AWS Secrets Manager tutorial

Topik

- Temukan rahasia yang tidak dilindungi dalam kode Anda dengan Amazon Reviewer CodeGuru
- Pindahkan rahasia hardcode ke AWS Secrets Manager
- Pindahkan kredensi database hardcode ke AWS Secrets Manager
- Buat rahasia klien Salesforce
- Siapkan rotasi pengguna bergantian untuk AWS Secrets Manager
- Siapkan rotasi pengguna tunggal untuk AWS Secrets Manager

Temukan rahasia yang tidak dilindungi dalam kode Anda dengan Amazon Reviewer CodeGuru

Amazon CodeGuru Reviewer adalah layanan yang menggunakan analisis program dan pembelajaran mesin untuk mendeteksi potensi cacat yang sulit ditemukan oleh pengembang dan menawarkan saran untuk meningkatkan kode Java dan Python Anda. CodeGuru Reviewer terintegrasi dengan Secrets Manager untuk menemukan rahasia yang tidak dilindungi dalam kode Anda. Untuk jenis rahasia yang dapat ditemukan, lihat Jenis rahasia yang terdeteksi oleh CodeGuru Reviewer di Panduan Pengguna Amazon CodeGuru Reviewer.

Setelah Anda menemukan rahasia hardcode, ambil tindakan untuk menggantinya:

- the section called “Ganti kredensi DB hardcode ”
- the section called “Ganti rahasia hardcode ”

Pindahkan rahasia hardcode ke AWS Secrets Manager

Jika Anda memiliki rahasia plaintext dalam kode Anda, kami sarankan Anda memutarnya dan menyimpannya di Secrets Manager. Memindahkan rahasia ke Secrets Manager memecahkan masalah rahasia yang terlihat oleh siapa saja yang melihat kode, karena ke depan, kode Anda mengambil rahasia langsung dari Secrets Manager. Memutar rahasia mencabut rahasia hardcode saat ini sehingga tidak lagi valid.

Untuk rahasia kredenal basis data, lihatPindahkan kredensi database hardcode ke AWS Secrets Manager.

Sebelum Anda mulai, Anda perlu menentukan siapa yang membutuhkan akses ke rahasia. Sebaiknya gunakan dua peran IAM untuk mengelola izin rahasia Anda:

- Peran yang mengelola rahasia dalam organisasi Anda. Untuk informasi selengkapnya, lihat the section called “Izin administrator Secrets Manager”. Anda akan membuat dan memutar rahasia menggunakan peran ini.
- Peran yang dapat menggunakan rahasia saat runtime, misalnya dalam tutorial ini yang Anda gunakan`RoleToRetrieveSecretAtRuntime`. Kode Anda mengasumsikan peran ini untuk mengambil rahasia. Dalam tutorial ini, Anda memberikan peran hanya izin untuk mengambil satu nilai rahasia, dan Anda memberikan izin dengan menggunakan kebijakan sumber daya rahasia. Untuk alternatif lain, lihatthe section called “Langkah selanjutnya”.

Langkah:

- Langkah 1: Buat Rahasianya
- Langkah 2: Perbarui kode Anda
- Langkah 3: Perbarui rahasianya
- Langkah selanjutnya

Langkah 1: Buat Rahasianya

Langkah pertama adalah menyalin rahasia hardcode yang ada ke Secrets Manager. Jika rahasianya terkait dengan AWS sumber daya, simpan di Wilayah yang sama dengan sumber daya. Jika tidak, simpan di Wilayah yang memiliki latensi terendah untuk kasus penggunaan Anda.

Untuk membuat rahasia (konsol)

1. Buka konsol Secrets Manager di https://console.aws.amazon.com/secretsmanager/.
2. Pilih Simpan rahasia baru.
3. Pada halaman Pilih jenis rahasia, lakukan hal berikut:
    a. Untuk Tipe rahasia, pilih Tipe rahasia lainnya.
    b. Masukkan rahasia Anda sebagai Key/value pasangan atau di Plaintext. Beberapa contoh:

API key

Masukkan sebagai key/value pasangan:

ClientID : *my_client_id*

ClientSecret : *wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY*

OAuth token

Masukkan sebagai plaintext:

AKIAI44QH8DHBEXAMPLE

Digital certificate

Masukkan sebagai plaintext:

```
-----BEGIN CERTIFICATE-----
EXAMPLE
-----END CERTIFICATE-----
```

Private key

Masukkan sebagai plaintext:

```
---- BEGIN PRIVATE KEY ----
EXAMPLE
---- END PRIVATE KEY ----
```

c. Untuk kunci Enkripsi, pilih aws/secretsmanageruntuk menggunakan Secrets Manager. Kunci yang dikelola AWS Tidak ada biaya untuk menggunakan kunci ini. Anda juga dapat menggunakan kunci yang dikelola pelanggan Anda sendiri, misalnya untuk mengakses rahasia dari yang lain Akun AWS. Untuk informasi tentang biaya penggunaan kunci yang dikelola pelanggan, lihatHarga.

d. Pilih Berikutnya.

4. Pada halaman Pilih jenis rahasia, lakukan hal berikut:

a. Masukkan nama Rahasia deskriptif dan Deskripsi.

b. Di Izin sumber daya, pilih Edit izin. Tempel kebijakan berikut, yang memungkinkan *RoleToRetrieveSecretAtRuntime* untuk mengambil rahasia, lalu pilih Simpan.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS":
 "arn:aws:iam::111122223333:role/RoleToRetrieveSecretAtRuntime"
            },
            "Action": "secretsmanager:GetSecretValue",
            "Resource": "*"
        }
    ]
}
```

c. Di bagian bawah halaman, pilih Selanjutnya.

5. Pada halaman Konfigurasi rotasi, matikan rotasi. Pilih Berikutnya.
6. Pada halaman Ulasan, tinjau detail rahasia Anda, lalu pilih Store.

Langkah 2: Perbarui kode Anda

Kode Anda harus mengambil peran IAM *RoleToRetrieveSecretAtRuntime* untuk dapat mengambil rahasia. Untuk informasi selengkapnya, lihat Beralih ke peran IAM (AWS API).

Selanjutnya, Anda memperbarui kode Anda untuk mengambil rahasia dari Secrets Manager menggunakan kode contoh yang disediakan oleh Secrets Manager.

Untuk menemukan kode sampel

1. Buka konsol Secrets Manager di https://console.aws.amazon.com/secretsmanager/.
2. Pada halaman Rahasia, pilih rahasia Anda.
3. Gulir ke bawah ke kode Contoh. Pilih bahasa pemrograman Anda, lalu salin cuplikan kode.

Dalam aplikasi Anda, hapus rahasia hardcode dan tempel cuplikan kode. Bergantung pada bahasa kode Anda, Anda mungkin perlu menambahkan panggilan ke fungsi atau metode dalam cuplikan.

Uji apakah aplikasi Anda berfungsi seperti yang diharapkan dengan rahasia menggantikan rahasia hardcode.

Langkah 3: Perbarui rahasianya

Langkah terakhir adalah mencabut dan memperbarui rahasia hardcode. Lihat sumber rahasia untuk menemukan instruksi untuk mencabut dan memperbarui rahasia. Misalnya, Anda mungkin perlu menonaktifkan rahasia saat ini dan menghasilkan rahasia baru.

Untuk memperbarui rahasia dengan nilai baru

1. Buka konsol Secrets Manager di https://console.aws.amazon.com/secretsmanager/.
2. Pilih Rahasia, lalu pilih rahasianya.
3. Pada halaman Detail rahasia, gulir ke bawah dan pilih Ambil nilai rahasia, lalu pilih Edit.
4. Perbarui rahasianya lalu pilih Simpan.

Selanjutnya, uji apakah aplikasi Anda berfungsi seperti yang diharapkan dengan rahasia baru.

Langkah selanjutnya

Setelah Anda menghapus rahasia hardcode dari kode Anda, beberapa ide untuk dipertimbangkan selanjutnya:

- Untuk menemukan rahasia hardcode di aplikasi Java dan Python Anda, kami merekomendasikan Amazon Reviewer. CodeGuru
- Anda dapat meningkatkan kinerja dan mengurangi biaya dengan menyimpan rahasia. Untuk informasi selengkapnya, lihat Dapatkan rahasia.
- Untuk rahasia yang Anda akses dari beberapa Wilayah, pertimbangkan untuk mereplikasi rahasia Anda untuk meningkatkan latensi. Untuk informasi selengkapnya, lihat Multi-region replikasi.
- Dalam tutorial ini, Anda *`RoleToRetrieveSecretAtRuntime`* hanya memberikan izin untuk mengambil nilai rahasia. Untuk memberikan peran lebih banyak izin, misalnya untuk mendapatkan metadata tentang rahasia atau untuk melihat daftar rahasia, lihat. the section called “Resource-based kebijakan”
- Dalam tutorial ini, Anda diberikan izin untuk *`RoleToRetrieveSecretAtRuntime`* menggunakan kebijakan sumber daya rahasia. Untuk cara lain untuk memberikan izin, lihatthe section called “Identity-based kebijakan”.

Pindahkan kredensi database hardcode ke AWS Secrets Manager

Jika Anda memiliki kredensyal database plaintext dalam kode Anda, kami sarankan Anda memindahkan kredensialnya ke Secrets Manager dan kemudian segera memutarnya. Memindahkan kredensyal ke Secrets Manager memecahkan masalah kredensyal yang terlihat oleh siapa saja yang melihat kode, karena ke depan, kode Anda mengambil kredensyal langsung dari Secrets Manager. Memutar rahasia memperbarui kata sandi dan kemudian mencabut kata sandi hardcode saat ini sehingga tidak lagi valid.

Untuk database Amazon RDS, Amazon Redshift, dan Amazon DocumentDB, gunakan langkah-langkah di halaman ini untuk memindahkan kredensyal hardcode ke Secrets Manager. Untuk jenis kredensi dan rahasia lainnya, lihat. the section called “Ganti rahasia hardcode ”

Sebelum Anda mulai, Anda perlu menentukan siapa yang membutuhkan akses ke rahasia. Sebaiknya gunakan dua peran IAM untuk mengelola izin rahasia Anda:

- Peran yang mengelola rahasia dalam organisasi Anda. Untuk informasi selengkapnya, lihat the section called “Izin administrator Secrets Manager”. Anda akan membuat dan memutar rahasia menggunakan peran ini.
- Peran yang dapat menggunakan kredensi saat runtime, `RoleToRetrieveSecretAtRuntime` dalam tutorial ini. Kode Anda mengasumsikan peran ini untuk mengambil rahasia.

Langkah:

- Langkah 1: Buat Rahasianya
- Langkah 2: Perbarui kode Anda
- Langkah 3: Putar rahasianya
- Langkah selanjutnya

Langkah 1: Buat Rahasianya

Langkah pertama adalah menyalin kredensi hardcode yang ada menjadi rahasia di Secrets Manager. Untuk latensi terendah, simpan rahasia di Wilayah yang sama dengan database.

Untuk membuat rahasia

1. Buka konsol Secrets Manager di https://console.aws.amazon.com/secretsmanager/.

2. Pilih Simpan rahasia baru.
3. Pada halaman Pilih jenis rahasia, lakukan hal berikut:
   a. Untuk jenis Rahasia, pilih jenis kredenal database untuk disimpan:
      - Basis data Amazon RDS
      - Basis data Amazon DocumentDB
      - Gudang data Amazon Redshift.
      - Untuk jenis rahasia lainnya, lihat Mengganti rahasia hardcode.
   b. Untuk Credentials, masukkan kredensi hardcode yang ada untuk database.
   c. Untuk kunci Enkripsi, pilih aws/secretsmanageruntuk menggunakan Secrets Manager. Kunci yang dikelola AWS Tidak ada biaya untuk menggunakan kunci ini. Anda juga dapat menggunakan kunci yang dikelola pelanggan Anda sendiri, misalnya untuk mengakses rahasia dari yang lain Akun AWS. Untuk informasi tentang biaya penggunaan kunci yang dikelola pelanggan, lihatHarga.
   d. Untuk Database, pilih database Anda.
   e. Pilih Berikutnya.
4. Pada halaman Konfigurasi rahasia, lakukan hal berikut:
   a. Masukkan nama Rahasia deskriptif dan Deskripsi.
   b. Di Izin sumber daya, pilih Edit izin. Tempel kebijakan berikut, yang memungkinkan *RoleToRetrieveSecretAtRuntime* untuk mengambil rahasia, lalu pilih Simpan.

      JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS":
 "arn:aws:iam::111122223333:role/RoleToRetrieveSecretAtRuntime"
            },
            "Action": "secretsmanager:GetSecretValue",
            "Resource": "*"
        }
```

```
    ]
}
```

c. Di bagian bawah halaman, pilih Selanjutnya.

5. Pada halaman Konfigurasi rotasi, matikan rotasi untuk saat ini. Anda akan menyalakannya nanti. Pilih Berikutnya.
6. Pada halaman Ulasan, tinjau detail rahasia Anda, lalu pilih Store.

Langkah 2: Perbarui kode Anda

Kode Anda harus mengambil peran IAM *RoleToRetrieveSecretAtRuntime* untuk dapat mengambil rahasia. Untuk informasi selengkapnya, lihat Beralih ke peran IAM (AWS API).

Selanjutnya, Anda memperbarui kode Anda untuk mengambil rahasia dari Secrets Manager menggunakan kode contoh yang disediakan oleh Secrets Manager.

Untuk menemukan kode sampel

1. Buka konsol Secrets Manager di https://console.aws.amazon.com/secretsmanager/.
2. Pada halaman Rahasia, pilih rahasia Anda.
3. Gulir ke bawah ke kode Contoh. Pilih bahasa Anda, lalu salin cuplikan kode.

Dalam aplikasi Anda, hapus kredensi hardcode dan tempel cuplikan kode. Bergantung pada bahasa kode Anda, Anda mungkin perlu menambahkan panggilan ke fungsi atau metode dalam cuplikan.

Uji apakah aplikasi Anda berfungsi seperti yang diharapkan dengan rahasia menggantikan kredensi hardcode.

Langkah 3: Putar rahasianya

Langkah terakhir adalah mencabut kredensi hardcode dengan memutar rahasianya. Rotasi adalah proses memperbarui rahasia secara berkala. Saat Anda memutar rahasia, Anda memperbarui kredensialnya di rahasia dan database. Secrets Manager dapat secara otomatis memutar rahasia untuk Anda pada jadwal yang Anda tetapkan.

Bagian dari pengaturan rotasi adalah memastikan bahwa fungsi rotasi Lambda dapat mengakses Secrets Manager dan database Anda. Saat Anda mengaktifkan rotasi otomatis, Secrets Manager membuat fungsi rotasi Lambda di VPC yang sama dengan database Anda sehingga memiliki akses

jaringan ke database. Fungsi rotasi Lambda juga harus dapat melakukan panggilan ke Secrets Manager untuk memperbarui rahasia. Kami menyarankan Anda membuat titik akhir Secrets Manager di VPC sehingga panggilan dari Lambda ke Secrets Manager tidak meninggalkan infrastruktur. AWS Untuk petunjuk, lihat the section called “Titik akhir VPC (AWS PrivateLink)”.

Untuk mengaktifkan rotasi

1. Buka konsol Secrets Manager di https://console.aws.amazon.com/secretsmanager/.
2. Pada halaman Rahasia, pilih rahasia Anda.
3. Pada halaman Detail rahasia, di bagian konfigurasi Rotasi, pilih Edit rotasi.
4. Dalam kotak dialog Edit konfigurasi rotasi, lakukan hal berikut:
    a. Nyalakan Rotasi otomatis.
    b. Di bawah jadwal Rotasi, masukkan jadwal Anda di zona waktu UTC.
    c. Pilih Putar segera ketika rahasia disimpan untuk memutar rahasia Anda ketika Anda menyimpan perubahan Anda.
    d. Di bawah fungsi Rotasi, pilih Buat fungsi Lambda baru dan masukkan nama untuk fungsi baru Anda. Secrets Manager menambahkan SecretsManager "" ke awal nama fungsi Anda.
    e. Untuk strategi Rotasi, pilih Single user.
    f. Pilih Simpan.

Untuk memeriksa bahwa rahasianya diputar

1. Buka konsol Secrets Manager di https://console.aws.amazon.com/secretsmanager/.
2. Pilih Rahasia, lalu pilih rahasianya.
3. Pada halaman Detail rahasia, gulir ke bawah dan pilih Ambil nilai rahasia.

    Jika nilai rahasia berubah, maka rotasi berhasil. Jika nilai rahasia tidak berubah, Anda perlu Memecahkan masalah rotasi melihat CloudWatch Log untuk fungsi rotasi.

Uji apakah aplikasi Anda berfungsi seperti yang diharapkan dengan rahasia yang diputar.

Langkah selanjutnya

Setelah Anda menghapus rahasia hardcode dari kode Anda, beberapa ide untuk dipertimbangkan selanjutnya:

- Anda dapat meningkatkan kinerja dan mengurangi biaya dengan menyimpan rahasia. Untuk informasi selengkapnya, lihat Dapatkan rahasia.
- Anda dapat memilih jadwal rotasi yang berbeda. Untuk informasi selengkapnya, lihat the section called “Jadwal rotasi”.
- Untuk menemukan rahasia hardcode di aplikasi Java dan Python Anda, kami merekomendasikan Amazon Reviewer. CodeGuru

Buat rahasia klien Salesforce

Dalam tutorial ini, Anda membuat rahasia klien Salesforce dan menyimpannya AWS Secrets Manager sebagai rahasia eksternal yang dikelola, mengaktifkan rotasi terkelola, dan kemudian menguji kredenal yang diputar dengan menukarnya dengan token akses. Tutorial ini membutuhkan waktu sekitar 20-30 menit untuk menyelesaikannya.

Tutorial ini menggunakan AWS layanan yang mungkin dikenakan biaya untuk Anda Akun AWS, termasuk biaya untuk rahasia yang Anda simpan di Secrets Manager dan, jika Anda memilih untuk menggunakannya, kunci yang dikelola pelanggan. Untuk informasi selengkapnya, lihat Harga.

> **Note**
>
> Tutorial ini menggunakan tipe SalesforceClientSecretrahasia sebagai contoh. Jenis rahasia eksternal terkelola lainnya mengikuti aliran serupa. Anda membuat rahasia menggunakan format yang ditentukan oleh mitra integrasi, mengaktifkan rotasi terkelola, dan kemudian menggunakan kredenal yang diputar. Untuk daftar mitra onboard dan format rahasia yang diperlukan, lihat. Mitra Integrasi

Sebelum Anda memulai tutorial ini, pastikan Anda memiliki yang berikut ini.

- Akun Salesforce (pengembang atau lainnya) yang memungkinkan Anda membuat Aplikasi Klien Eksternal (ECA).
- Izin administratif untuk Anda Akun AWS. Dalam pengaturan produksi, ini adalah praktik terbaik untuk menggunakan peran yang berbeda untuk tugas yang berbeda. Misalnya, peran yang mengelola rahasia di organisasi Anda akan menciptakan rahasia, dan peran terpisah akan menggunakan rahasia saat runtime. Untuk langkah-langkah tutorial, kami sarankan Anda terus menggunakan identitas yang sama. Untuk informasi tentang cara mengatur izin di lingkungan produksi, lihatthe section called “Kontrol autentikasi dan akses”.

Tutorial ini mencakup langkah-langkah berikut:

- Langkah 1: Buat Aplikasi Klien Eksternal dan aktifkan OAuth
- Langkah 2: Buat rahasia eksternal yang dikelola dan aktifkan rotasi
- Langkah 3: Uji kredensi yang diputar
- Langkah 4: Bersihkan Sumber Daya
- Langkah selanjutnya

Langkah 1: Buat Aplikasi Klien Eksternal dan aktifkan OAuth

Langkah pertama adalah membuat External Client App (ECA) di Salesforce dan mengkonfigurasinya untuk OAuth sehingga Secrets Manager dapat memutar rahasia klien. Untuk prosedur Salesforce lengkap, lihat Mengatur otorisasi dengan Aplikasi Klien Eksternal dalam dokumentasi Salesforce.

Saat Anda mengatur ECA, pastikan Anda mengonfigurasi yang berikut:

- Di tingkat organisasi, aktifkan Izinkan akses ke rahasia konsumen Aplikasi Klien Eksternal melalui REST API. Pengaturan ini memungkinkan Secrets Manager untuk mengambil dan memutar rahasia konsumen melalui Salesforce REST API.
- Di ECA, aktifkan lingkup OAuth Kelola data pengguna melalui API (api).
- Pada ECA, aktifkan Aktifkan Aliran Kredensial Klien.
- Pada ECA, aktifkan Aktifkan Token Exchange Flow dan Memerlukan rahasia untuk Token Exchange Flow.

Setelah Anda membuat ECA, kumpulkan nilai-nilai yang perlu Anda simpan dalam rahasia di langkah berikutnya:

- Kunci konsumen (ID klien) dan rahasia konsumen (rahasia klien) dari pengaturan ECA Manager OAuth.
- URL dasar Salesforce My Domain Anda, misalnya. `https://MyDomainName.my.salesforce.com`
- ID aplikasi dan ID konsumen untuk ECA. Ini sesuai dengan `external_client_app_identifier` dan `consumer_id` bidang dalam dokumentasi Salesforce. Untuk mempelajari cara mengambil nilai-nilai ini, lihat Stage OAuth Credentials dalam dokumentasi Salesforce.

Untuk deskripsi setiap bidang yang diperlukan dalam rahasia, lihatRahasia Klien Salesforce.

Langkah 2: Buat rahasia eksternal yang dikelola dan aktifkan rotasi

Selanjutnya, Anda menyimpan rahasia klien Salesforce di Secrets Manager sebagai rahasia eksternal yang dikelola dan mengaktifkan rotasi terkelola. Dengan rotasi terkelola, Anda tidak membuat atau mempertahankan fungsi rotasi Lambda. Secrets Manager memutar rahasia untuk Anda.

Untuk membuat rahasia dan mengaktifkan rotasi (konsol)

1. Buka konsol Secrets Manager di https://console.aws.amazon.com/secretsmanager/.
2. Pilih Simpan rahasia baru.
3. Pada halaman Pilih jenis rahasia, lakukan hal berikut:
    a. Untuk tipe Rahasia, pilih jenis rahasia eksternal terkelola rahasia klien Salesforce () SalesforceClientSecret.
    b. Masukkan bidang nilai rahasia untuk rahasia klien Salesforce: kunci konsumen, rahasia konsumen, URI dasar, ID aplikasi, dan ID konsumen. Untuk deskripsi setiap bidang, lihatRahasia Klien Salesforce.
    c. Untuk kunci Enkripsi, pilih aws/secretsmanageruntuk menggunakan Secrets Manager. Kunci yang dikelola AWS Tidak ada biaya untuk menggunakan kunci ini. Anda juga dapat menggunakan kunci yang dikelola pelanggan Anda sendiri. Untuk informasi tentang biaya penggunaan kunci yang dikelola pelanggan, lihatHarga.
    d. Pilih Berikutnya.
4. Pada halaman Konfigurasi rahasia, masukkan nama Rahasia deskriptif dan Deskripsi, lalu pilih Berikutnya.
5. Pada halaman Konfigurasi rotasi, lakukan hal berikut. Untuk rahasia eksternal yang dikelola, rotasi diaktifkan secara default selama pembuatan rahasia.
    a. Konfirmasikan itu Rotasi otomatis dihidupkan.
    b. Di bawah Metadata rotasi, untuk versi API (`apiVersion`), masukkan versi API organisasi Salesforce Anda. Bidang ini wajib diisi. Itu harus dalam formatv*XX.X*, di mana *X* adalah karakter numerik, dan harus setidaknyav`65.0`. Jika Anda ingin Secrets Manager memutar rahasia menggunakan seperangkat kredensi administratif terpisah, berikan juga ARN rahasia admin. Untuk deskripsi setiap bidang metadata, lihat. Rahasia Klien Salesforce

c. Di Izin layanan untuk rotasi rahasia, pilih Buat peran baru untuk secara otomatis membuat peran IAM dengan izin yang dibutuhkan Secrets Manager untuk memutar rahasia. Secara default, izin dicakup ke mitra integrasi di AWS Wilayah tempat rahasia dibuat. Untuk informasi selengkapnya, lihat Keamanan dan izin.

d. Di bawah jadwal Rotasi, masukkan jadwal Anda di zona waktu UTC, misalnya putar secara otomatis setiap 30 hari.

e. Pilih Berikutnya.

6. Pada halaman Ulasan, tinjau detail rahasia Anda, lalu pilih Store.

Untuk lebih banyak cara mengonfigurasi rotasi untuk rahasia eksternal terkelola, termasuk AWS AWS CLI, lihatRotate Secrets Manager mengelola rahasia eksternal.

Langkah 3: Uji kredensi yang diputar

Langkah terakhir adalah mengonfirmasi bahwa kredenal yang diputar berfungsi dengan mengambil nilai rahasia dan menukar kredensialnya dengan token akses Salesforce.

Untuk mengambil nilai rahasia (konsol)

1. Buka konsol Secrets Manager di https://console.aws.amazon.com/secretsmanager/.
2. Pada halaman Rahasia, pilih rahasia Anda.
3. Pada halaman Detail rahasia, gulir ke bawah dan pilih Ambil nilai rahasia. Perhatikan arus `consumerKey` dan `consumerSecret` nilai.

> **Note**
>
> Konfirmasikan bahwa `consumerSecret` nilai `consumerKey` dan sebenarnya telah berubah dari nilai yang awalnya Anda simpan. Jika belum berubah, tunggu beberapa menit dan periksa lagi. Jika masih belum berubah, periksa AWS CloudTrail rahasia `RotationFailed` peristiwa apa pun, yang mencakup detail tentang mengapa rotasi gagal. Untuk informasi selengkapnya, lihat Rahasia eksternal saya yang dikelola tidak berputar.

Selanjutnya, gunakan kredensi yang diambil untuk meminta token akses dari Salesforce. Ganti *`MyDomainName`* dengan nama Domain Saya Salesforce Anda, dan ganti *`consumerKey`* dan *`consumerSecret`* dengan nilai yang Anda ambil.

```
curl -X POST "https://MyDomainName.my.salesforce.com/services/oauth2/token" \
  -H "Content-Type: application/x-www-form-urlencoded" \
  -d "grant_type=client_credentials" \
  -d "client_id=consumerKey" \
  -d "client_secret=consumerSecret"
```

Jika kredensialnya valid, Salesforce mengembalikan respons JSON yang berisi file. `access_token` Menerima token akses mengonfirmasi bahwa kredenal yang diputar berfungsi.

Saat Anda memasukkan perintah di shell perintah, ada risiko riwayat perintah diakses atau utilitas memiliki akses ke parameter perintah Anda. Lihat the section called “Mengurangi risiko menggunakan AWS CLI untuk menyimpan rahasia Anda AWS Secrets Manager”.

Langkah 4: Bersihkan Sumber Daya

Untuk menghindari biaya yang sedang berlangsung, hapus sumber daya yang Anda buat dalam tutorial ini saat Anda tidak lagi membutuhkannya: rahasia dan peran rotasi IAM yang dibuat Secrets Manager. Menghapus rahasia juga menghentikan rotasi otomatis.

Untuk menghapus rahasia (konsol)

1. Buka konsol Secrets Manager di https://console.aws.amazon.com/secretsmanager/.
2. Pada halaman Rahasia, pilih rahasia Anda.
3. Di bagian Detail rahasia, pilih Tindakan, lalu pilih Hapus rahasia.
4. Dalam kotak dialog Nonaktifkan rahasia dan jadwal penghapusan, masukkan masa tunggu, lalu pilih Jadwal penghapusan. Secrets Manager membuat rahasia tidak dapat diakses segera dan menjadwalkannya untuk dihapus setelah jendela pemulihan. Untuk informasi selengkapnya, lihat Hapus sebuah AWS Secrets Manager Rahasia.

Untuk menghapus peran rotasi (konsol)

1. Buka konsol IAM di https://console.aws.amazon.com/iam/, lalu pilih Peran.
2. Temukan peran yang dibuat Secrets Manager untuk rotasi, pilih, lalu pilih Hapus. Untuk informasi lebih lanjut, lihat Menghapus peran atau profil instans dalam Panduan Pengguna IAM.

Langkah selanjutnya

- Untuk mempelajari lebih lanjut tentang rahasia eksternal terkelola dan mitra Secrets Manager terintegrasi dengan Secrets Manager, lihatPenggunaan AWS Secrets Manager rahasia eksternal yang dikelola untuk mengelola rahasia Pihak Ketiga.
- Untuk meninjau nilai rahasia yang diperlukan dan bidang metadata rotasi untuk Salesforce, lihat. Rahasia Klien Salesforce
- Untuk lebih banyak cara mengatur rotasi untuk rahasia eksternal yang dikelola, lihatPutar rahasia eksternal yang dikelola.
- Anda dapat meningkatkan kinerja dan mengurangi biaya dengan menyimpan rahasia. Untuk informasi selengkapnya, lihat Dapatkan rahasia.
- Untuk mempelajari tentang biaya menggunakan Secrets Manager, lihatHarga.

Siapkan rotasi pengguna bergantian untuk AWS Secrets Manager

Dalam tutorial ini, Anda belajar cara mengatur rotasi pengguna bergantian untuk rahasia yang berisi kredensi database. Rotasi pengguna bergantian adalah strategi rotasi di mana Secrets Manager mengkloning pengguna dan kemudian mengganti kredensi pengguna mana yang diperbarui. Strategi ini adalah pilihan yang baik jika Anda membutuhkan ketersediaan tinggi untuk rahasia Anda, karena salah satu pengguna bolak-balik memiliki kredensi saat ini ke database sementara yang lain sedang diperbarui. Untuk informasi selengkapnya, lihat the section called “Pengguna bergantian”.

Untuk mengatur rotasi pengguna bergantian, Anda memerlukan dua rahasia:

- Satu rahasia dengan kredensil yang ingin Anda putar.
- Rahasia kedua yang memiliki kredensi admin.

  Pengguna ini memiliki izin untuk mengkloning pengguna pertama dan mengubah kata sandi pengguna pertama. Dalam tutorial ini, Anda memiliki Amazon RDS membuat rahasia ini untuk pengguna admin. Amazon RDS juga mengelola rotasi kata sandi admin. Untuk informasi selengkapnya, lihat the section called “Rotasi terkelola”.

Bagian pertama dari tutorial ini adalah menyiapkan lingkungan yang realistis. Untuk menunjukkan cara kerja rotasi, tutorial ini menggunakan contoh database Amazon RDS MySQL. Untuk keamanan, database berada dalam VPC yang membatasi akses internet masuk. Untuk terhubung ke database dari komputer lokal Anda melalui internet, Anda menggunakan bastion host, server di VPC yang

dapat terhubung ke database, tetapi itu juga memungkinkan koneksi SSH dari internet. Host bastion dalam tutorial ini adalah instans Amazon EC2, dan grup keamanan untuk instance mencegah jenis koneksi lainnya.

Setelah Anda menyelesaikan tutorial, kami sarankan Anda membersihkan sumber daya dari tutorial. Jangan menggunakannya dalam pengaturan produksi.

Rotasi Secrets Manager menggunakan AWS Lambda fungsi untuk memperbarui rahasia dan database. Untuk informasi tentang biaya penggunaan fungsi Lambda, lihat. Harga

Tutorial:

- Izin
- Prasyarat
- Langkah 1: Buat pengguna database Amazon RDS
- Langkah 2: Buat rahasia untuk kredensi pengguna
- Langkah 3: Uji rahasia yang diputar
- Langkah 4: Bersihkan Sumber Daya
- Langkah selanjutnya

Izin

Untuk prasyarat tutorial, Anda memerlukan izin administratif untuk Anda. Akun AWS Dalam pengaturan produksi, ini adalah praktik terbaik untuk menggunakan peran yang berbeda untuk setiap langkah. Misalnya, peran dengan izin admin database akan membuat database Amazon RDS, dan peran dengan izin admin jaringan akan mengatur VPC dan grup keamanan. Untuk langkah-langkah tutorial, kami sarankan Anda terus menggunakan identitas yang sama.

Untuk informasi tentang cara mengatur izin di lingkungan produksi, lihatthe section called “Kontrol autentikasi dan akses”.

Prasyarat

Dalam tutorial ini, Anda akan melakukan langkah-langkah berikut:

- Prasyarat A: Amazon VPC
- Prereq B: Instans Amazon EC2
- Prereq C: Basis data Amazon RDS dan rahasia Secrets Manager untuk kredensi admin

- Prereq D: Izinkan komputer lokal Anda terhubung ke instans EC2

Prasyarat A: Amazon VPC

Pada langkah ini, Anda membuat VPC tempat Anda dapat meluncurkan database Amazon RDS dan instans Amazon EC2. Pada langkah selanjutnya, Anda akan menggunakan komputer Anda untuk terhubung melalui internet ke benteng dan kemudian ke database, jadi Anda perlu mengizinkan lalu lintas keluar dari VPC. Untuk melakukan ini, Amazon VPC melampirkan gateway internet ke VPC dan menambahkan rute di tabel rute sehingga lalu lintas yang ditujukan untuk di luar VPC dikirim ke gateway internet.

Dalam VPC, Anda membuat titik akhir Secrets Manager dan endpoint Amazon RDS. Saat Anda mengatur rotasi otomatis di langkah selanjutnya, Secrets Manager membuat fungsi rotasi Lambda di dalam VPC sehingga dapat mengakses database. Fungsi rotasi Lambda juga memanggil Secrets Manager untuk memperbarui rahasia, dan memanggil Amazon RDS untuk mendapatkan informasi koneksi database. Dengan membuat titik akhir dalam VPC, Anda memastikan bahwa panggilan dari fungsi Lambda ke Secrets Manager dan Amazon RDS tidak meninggalkan infrastruktur. AWS Sebaliknya, mereka diarahkan ke titik akhir dalam VPC.

Untuk membuat VPC

1. Buka konsol Amazon VPC di. https://console.aws.amazon.com/vpc/
2. Pilih Buat VPC.
3. Pada halaman Buat VPC, pilih VPC dan lainnya.
4. Di bawah Generasi otomatis tag nama, di bawah Auto-generate, masukkan**`SecretsManagerTutorial`**.
5. Untuk opsi DNS, pilih keduanya **`Enable DNS hostnames`** dan**`Enable DNS resolution`**.
6. Pilih Buat VPC.

Untuk membuat titik akhir Secrets Manager di dalam VPC

1. Di konsol Amazon VPC, di bawah Endpoints, pilih Create Endpoint.
2. Di bawah Pengaturan titik akhir, untuk Nama, masukkan**`SecretsManagerTutorialEndpoint`**.

3. Di bawah Layanan, masukkan **`secretsmanager`** untuk memfilter daftar, lalu pilih titik akhir Secrets Manager di bagian Anda Wilayah AWS. Misalnya, di AS Timur (Virginia N.), pilih`com.amazonaws.us-east-1.secretsmanager`.
4. Untuk VPC, pilih. **`vpc**** (SecretsManagerTutorial)`**
5. Untuk Subnet, pilih semua Availability Zone, dan kemudian untuk masing-masing Subnet, pilih Subnet ID untuk disertakan.
6. Untuk jenis alamat IP, pilih**`IPv4`**.
7. Untuk grup Keamanan, pilih grup keamanan default.
8. Untuk Kebijakan, pilih**`Full access`**.
9. Pilih Buat titik akhir.

Untuk membuat endpoint Amazon RDS dalam VPC

1. Di konsol Amazon VPC, di bawah Endpoints, pilih Create Endpoint.
2. Di bawah Pengaturan titik akhir, untuk Nama, masukkan**`RDSTutorialEndpoint`**.
3. Di bawah Layanan, masukkan **`rds`** untuk memfilter daftar, lalu pilih titik akhir Amazon RDS di Anda. Wilayah AWS Misalnya, di AS Timur (Virginia N.), pilih`com.amazonaws.us-east-1.rds`.
4. Untuk VPC, pilih. **`vpc**** (SecretsManagerTutorial)`**
5. Untuk Subnet, pilih semua Availability Zone, dan kemudian untuk masing-masing Subnet, pilih Subnet ID untuk disertakan.
6. Untuk jenis alamat IP, pilih**`IPv4`**.
7. Untuk grup Keamanan, pilih grup keamanan default.
8. Untuk Kebijakan, pilih**`Full access`**.
9. Pilih Buat titik akhir.

Prereq B: Instans Amazon EC2

Basis data Amazon RDS yang Anda buat di langkah selanjutnya akan berada di VPC, jadi untuk mengaksesnya, Anda memerlukan host benteng. Host bastion juga ada di VPC, tetapi pada langkah selanjutnya, Anda mengonfigurasi grup keamanan untuk memungkinkan komputer lokal Anda terhubung ke host bastion dengan SSH.

Untuk membuat instance EC2 untuk host bastion

1. Buka konsol Amazon EC2 di. https://console.aws.amazon.com/ec2/
2. Pilih Instans dan kemudian pilih Launch Instances.
3. Pada Nama dan tag, untuk Nama, masukkan **`SecretsManagerTutorialInstance`**.
4. Di bawah Application dan OS Images, pertahankan default**`Amazon Linux 2 AMI (HMV) Kernel 5.10`**.
5. Di bawah tipe Instance, pertahankan default**`t2.micro`**.
6. Di bawah Key pair, pilih Create key pair.

   Dalam kotak dialog Create key pair, untuk nama Key pair, masukkan**`SecretsManagerTutorialKeyPair`**, lalu pilih Create key pair.

   Key pair diunduh secara otomatis.
7. Di bawah Pengaturan jaringan, pilih Edit, lalu lakukan hal berikut:
   a. Untuk VPC, pilih. **`vpc-**** SecretsManagerTutorial`**
   b. Untuk IP Auto-assign Publik, pilih**`Enable`**.
   c. Untuk Firewall, pilih Pilih grup keamanan yang ada.
   d. Untuk grup keamanan umum, pilih**`default`**.
8. Pilih Luncurkan instans.

Prereq C: Basis data Amazon RDS dan rahasia Secrets Manager untuk kredensi admin

Pada langkah ini, Anda membuat database Amazon RDS MySQL dan mengonfigurasinya sehingga Amazon RDS membuat rahasia untuk memuat kredensi admin. Kemudian Amazon RDS secara otomatis mengelola rotasi rahasia admin untuk Anda. Untuk informasi selengkapnya, lihat Rotasi terkelola.

Sebagai bagian dari pembuatan database Anda, Anda menentukan host bastion yang Anda buat pada langkah sebelumnya. Kemudian Amazon RDS menyiapkan grup keamanan sehingga database dan instance dapat saling mengakses. Anda menambahkan aturan ke grup keamanan yang dilampirkan ke instance untuk memungkinkan komputer lokal Anda terhubung dengannya juga.

Untuk membuat database Amazon RDS dengan rahasia Secrets Manager yang berisi kredensi admin

1. Di konsol Amazon RDS, pilih Buat database.
2. Di bagian Opsi mesin, untuk jenis mesin, pilih**MySQL**.
3. Di bagian Template, pilih**Free tier**.
4. Di bagian Pengaturan, lakukan hal berikut:
   a. Untuk Pengidentifikasi instans DB, masukkan **SecretsManagerTutorial**.
   b. Di bawah Pengaturan kredensi, pilih Kelola kredenal master di. AWS Secrets Manager
5. Di bagian Konektivitas, untuk sumber daya Komputer, pilih Connect ke sumber daya komputer EC2, dan kemudian untuk Instans EC2, pilih. **SecretsManagerTutorialInstance**
6. Pilih Buat basis data.

Prereq D: Izinkan komputer lokal Anda terhubung ke instans EC2

Pada langkah ini, Anda mengonfigurasi instans EC2 yang Anda buat di Prereq B untuk memungkinkan komputer lokal Anda terhubung dengannya. Untuk melakukan ini, Anda mengedit grup keamanan yang ditambahkan Amazon RDS di Prereq C untuk menyertakan aturan yang memungkinkan alamat IP komputer Anda terhubung dengan SSH. Aturan ini memungkinkan komputer lokal Anda (diidentifikasi oleh alamat IP Anda saat ini) untuk terhubung ke host bastion dengan menggunakan SSH melalui internet.

Untuk memungkinkan komputer lokal Anda terhubung ke instans EC2

1. Buka konsol Amazon EC2 di. https://console.aws.amazon.com/ec2/
2. Pada instans EC2 SecretsManagerTutorialInstance, pada tab Keamanan, di bawah Grup keamanan, pilih**sg-*** (ec2-rds-X)**.
3. Di bawah Aturan input, pilih Edit aturan masuk.
4. Pilih Tambah aturan, dan kemudian untuk aturan, lakukan hal berikut:
   a. Untuk Jenis, pilih **SSH**.
   b. Untuk tipe Sumber, pilih**My IP**.

Langkah 1: Buat pengguna database Amazon RDS

Pertama, Anda memerlukan pengguna yang kredensialnya akan disimpan dalam rahasia. Untuk membuat pengguna, masuk ke database Amazon RDS dengan kredensi admin. Untuk kesederhanaan, dalam tutorial, Anda membuat pengguna dengan izin penuh ke database. Dalam pengaturan produksi, ini tidak khas, dan kami menyarankan Anda mengikuti prinsip hak istimewa paling sedikit.

Untuk terhubung ke database, Anda menggunakan alat klien MySQL. Dalam tutorial ini, Anda menggunakan MySQL Workbench, sebuah aplikasi. GUI-based Untuk menginstal MySQL Workbench, lihat Download MySQL Workbench.

Untuk terhubung ke database, buat konfigurasi koneksi di MySQL Workbench. Untuk konfigurasi, Anda memerlukan beberapa informasi dari Amazon EC2 dan Amazon RDS.

Untuk membuat koneksi database di MySQL Workbench

1. Di MySQL Workbench, di sebelah MySQL Connections, pilih tombol (+).
2. Dalam kotak dialog Setup New Connection, lakukan hal berikut:
    a. Untuk Nama Koneksi, masukkan**`SecretsManagerTutorial`**.
    b. Untuk Metode Koneksi, pilih**`Standard TCP/IP over SSH`**.
    c. Pada tab Parameter, lakukan hal berikut:
        i. Untuk SSH Hostname, masukkan alamat IP publik instans Amazon EC2.

            Anda dapat menemukan alamat IP di konsol Amazon EC2 dengan memilih instans. SecretsManagerTutorialInstance Salin alamat IP di bawah DNS IPv4 Publik.
        ii. Untuk Nama Pengguna SSH, masukkan**`ec2-user`**.
        iii. Untuk SSH Keyfile, pilih file key pair yang SecretsManagerTutorialKeyPair.pemAnda unduh di prasyarat sebelumnya.
        iv. Untuk MySQL Hostname, masukkan alamat endpoint Amazon RDS.

            Anda dapat menemukan alamat endpoint pada konsol Amazon RDS dengan memilih instance database secretsmanagertutorialdb. Salin alamat di bawah Endpoint.
        v. Untuk Nama Pengguna, masukkan**`admin`**.
    d. Pilih OK.

Untuk mengambil kata sandi admin

1. Di konsol Amazon RDS, navigasikan ke database Anda.
2. Pada tab Konfigurasi, di bawah Master Credentials ARN, pilih Manage in Secrets Manager.

   Konsol Secrets Manager terbuka.
3. Di halaman detail rahasia, pilih Ambil nilai rahasia.
4. Kata sandi muncul di bagian Nilai rahasia.

Untuk membuat pengguna database

1. Di MySQL Workbench, pilih koneksi. SecretsManagerTutorial
2. Masukkan kata sandi admin yang Anda ambil dari rahasia.
3. Di MySQL Workbench, di jendela Query, masukkan perintah berikut (termasuk kata sandi yang kuat) dan kemudian pilih Execute. Fungsi rotasi menguji rahasia yang diperbarui dengan menggunakan SELECT, sehingga **`appuser`** harus memiliki hak istimewa itu minimal.

   ```
   CREATE DATABASE myDB;
   CREATE USER 'appuser'@'%' IDENTIFIED BY 'EXAMPLE-PASSWORD';
   GRANT SELECT ON myDB . * TO 'appuser'@'%';
   ```

   Di jendela Output, Anda melihat perintah berhasil.

Langkah 2: Buat rahasia untuk kredensi pengguna

Selanjutnya, Anda membuat rahasia untuk menyimpan kredensi pengguna yang baru saja Anda buat. Ini adalah rahasia yang akan Anda putar. Anda mengaktifkan rotasi otomatis, dan untuk menunjukkan strategi pengguna bergantian, Anda memilih rahasia superuser terpisah yang memiliki izin untuk mengubah kata sandi pengguna pertama.

1. Buka konsol Secrets Manager di https://console.aws.amazon.com/secretsmanager/.
2. Pilih Simpan rahasia baru.
3. Pada halaman Pilih jenis rahasia, lakukan hal berikut:

   a. Untuk jenis Rahasia, pilih Kredensial untuk database Amazon RDS.

   b. Untuk Kredensial, masukkan nama pengguna **`appuser`** dan kata sandi yang Anda masukkan untuk pengguna database yang Anda buat menggunakan MySQL Workbench.

   c. Untuk Database, pilih secretsmanagertutorialdb.

   d. Pilih Berikutnya.

4. Pada halaman Konfigurasi rahasia, untuk nama Rahasia, masukkan **`SecretsManagerTutorialAppuser`** dan kemudian pilih Berikutnya.

5. Pada halaman Konfigurasi rotasi, lakukan hal berikut:

   a. Nyalakan rotasi otomatis.

   b. Untuk jadwal Rotasi, atur jadwal Hari: **2** Hari dengan Durasi:**2h**. Tetap Putar segera dipilih.

   c. Untuk fungsi Rotasi, pilih Buat fungsi rotasi, dan kemudian untuk nama fungsi, masukkan**`tutorial-alternating-users-rotation`**.

   d. Untuk strategi Rotasi, pilih Alternating users, dan kemudian di bawah Admin credential secret, pilih rahasia bernama rds! kluster... yang memiliki Deskripsi yang menyertakan nama database yang Anda buat dalam tutorial ini**`secretsmanagertutorial`**, misalnya`Secret associated with primary RDS DB instance: arn:aws:rds:`*`Region`*`:`*`AccountId`*`:db:secretsmanagertutorial`.

   e. Pilih Berikutnya.

6. Pada halaman Review, pilih Store.

   Secrets Manager kembali ke halaman detail rahasia. Di bagian atas halaman, Anda dapat melihat status konfigurasi rotasi. Secrets Manager menggunakan CloudFormation untuk membuat sumber daya seperti fungsi rotasi Lambda dan peran eksekusi yang menjalankan fungsi Lambda. Setelah CloudFormation selesai, spanduk berubah menjadi Rahasia yang dijadwalkan untuk rotasi. Rotasi pertama selesai.

Langkah 3: Uji rahasia yang diputar

Sekarang rahasianya diputar, Anda dapat memeriksa apakah rahasia tersebut berisi kredensil baru yang valid. Kata sandi dalam rahasia telah berubah dari kredensi asli.

Untuk mengambil kata sandi baru dari rahasia

1. Buka konsol Secrets Manager di https://console.aws.amazon.com/secretsmanager/.

2. Pilih Rahasia, lalu pilih rahasianya**`SecretsManagerTutorialAppuser`**.

3. Pada halaman Detail rahasia, gulir ke bawah dan pilih Ambil nilai rahasia.
4. Dalam Key/valuetabel, salin nilai Rahasia untuk**`password`**.

Untuk menguji kredensialnya

1. Di MySQL Workbench, klik kanan koneksi dan kemudian pilih Edit SecretsManagerTutorialKoneksi.
2. Dalam kotak dialog Kelola Koneksi Server, untuk Nama Pengguna**`appuser`**, masukkan, lalu pilih Tutup.
3. Kembali di MySQL Workbench, pilih koneksi. SecretsManagerTutorial
4. Di kotak dialog Open SSH Connection, untuk Kata Sandi, tempel kata sandi yang Anda ambil dari rahasia, lalu pilih OK.

   Jika kredensialnya valid, maka MySQL Workbench terbuka ke halaman desain untuk database.

Ini menunjukkan bahwa rotasi rahasia berhasil. Kredensi dalam rahasia telah diperbarui dan itu adalah kata sandi yang valid untuk terhubung ke database.

Langkah 4: Bersihkan Sumber Daya

Jika Anda ingin mencoba strategi rotasi lain, rotasi pengguna tunggal, lewati pembersihan sumber daya dan bukathe section called “Rotasi pengguna tunggal”.

Jika tidak, untuk menghindari potensi biaya, dan untuk menghapus instans EC2 yang memiliki akses ke internet, hapus sumber daya berikut yang Anda buat dalam tutorial ini dan prasyaratnya:

- Contoh basis data Amazon RDS. Untuk petunjuknya, lihat Menghapus instans DB di Panduan Pengguna Amazon RDS.
- Instans Amazon EC2. Untuk petunjuknya, lihat Mengakhiri instance di Panduan Pengguna Amazon EC2.
- Rahasia Secrets Manager`SecretsManagerTutorialAppuser`. Untuk petunjuk, lihat the section called “Hapus rahasia”.
- Titik akhir Secrets Manager. Untuk petunjuk, lihat Menghapus titik akhir VPC di Panduan.AWS PrivateLink
- Titik akhir VPC. Untuk petunjuk, lihat Menghapus VPC Anda di Panduan.AWS PrivateLink

Langkah selanjutnya

- Pelajari cara mengambil rahasia di aplikasi Anda.
- Pelajari tentang jadwal rotasi lainnya.

Siapkan rotasi pengguna tunggal untuk AWS Secrets Manager

Dalam tutorial ini, Anda belajar cara mengatur rotasi pengguna tunggal untuk rahasia yang berisi kredensi database. Rotasi pengguna tunggal adalah strategi rotasi di mana Secrets Manager memperbarui kredensi pengguna baik dalam rahasia maupun database. Untuk informasi selengkapnya, lihat the section called “Pengguna tunggal”.

Setelah Anda menyelesaikan tutorial, kami sarankan Anda membersihkan sumber daya dari tutorial. Jangan menggunakannya dalam pengaturan produksi.

Rotasi Secrets Manager menggunakan AWS Lambda fungsi untuk memperbarui rahasia dan database. Untuk informasi tentang biaya penggunaan fungsi Lambda, lihat. Harga

Daftar Isi

- Izin
- Prasyarat
- Langkah 1: Buat pengguna database Amazon RDS
- Langkah 2: Buat rahasia untuk kredensi pengguna database
- Langkah 3: Uji kata sandi yang diputar
- Langkah 4: Bersihkan Sumber Daya
- Langkah selanjutnya

Izin

Untuk prasyarat tutorial, Anda memerlukan izin administratif untuk Anda. Akun AWS Dalam pengaturan produksi, ini adalah praktik terbaik untuk menggunakan peran yang berbeda untuk setiap langkah. Misalnya, peran dengan izin admin database akan membuat database Amazon RDS, dan peran dengan izin admin jaringan akan mengatur VPC dan grup keamanan. Untuk langkah-langkah tutorial, kami sarankan Anda terus menggunakan identitas yang sama.

Untuk informasi tentang cara mengatur izin di lingkungan produksi, lihatthe section called “Kontrol autentikasi dan akses”.

Prasyarat

Prasyarat untuk tutorial ini adalah. the section called “Rotasi pengguna bergantian” Jangan membersihkan sumber daya di akhir tutorial pertama. Setelah tutorial itu, Anda memiliki lingkungan yang realistis dengan database Amazon RDS dan rahasia Secrets Manager yang berisi kredensi admin untuk database. Anda juga memiliki rahasia kedua yang berisi kredensil untuk pengguna database, tetapi Anda tidak menggunakan rahasia itu dalam tutorial ini.

Anda juga memiliki koneksi yang dikonfigurasi di MySQL Workbench untuk terhubung ke database dengan kredensi admin.

Langkah 1: Buat pengguna database Amazon RDS

Pertama, Anda memerlukan pengguna yang kredensialnya akan disimpan dalam rahasia. Untuk membuat pengguna, masuk ke database Amazon RDS dengan kredensi admin yang disimpan dalam rahasia. Untuk kesederhanaan, dalam tutorial, Anda membuat pengguna dengan izin penuh ke database. Dalam pengaturan produksi, ini tidak khas, dan kami menyarankan Anda mengikuti prinsip hak istimewa paling sedikit.

Untuk mengambil kata sandi admin

1. Di konsol Amazon RDS, navigasikan ke database Anda.
2. Pada tab Konfigurasi, di bawah Master Credentials ARN, pilih Manage in Secrets Manager.

   Konsol Secrets Manager terbuka.
3. Di halaman detail rahasia, pilih Ambil nilai rahasia.
4. Kata sandi muncul di bagian Nilai rahasia.

Untuk membuat pengguna database

1. Di MySQL Workbench, klik kanan koneksi dan kemudian pilih Edit SecretsManagerTutorialKoneksi.
2. Dalam kotak dialog Kelola Koneksi Server, untuk Nama Pengguna**`admin`**, masukkan, lalu pilih Tutup.
3. Kembali di MySQL Workbench, pilih koneksi. SecretsManagerTutorial

4. Masukkan kata sandi admin yang Anda ambil dari rahasia.
5. Di MySQL Workbench, di jendela Query, masukkan perintah berikut (termasuk kata sandi yang kuat) dan kemudian pilih Execute. Fungsi rotasi menguji rahasia yang diperbarui dengan menggunakan SELECT, sehingga **dbuser** harus memiliki hak istimewa itu minimal.

```
CREATE USER 'dbuser'@'%' IDENTIFIED BY 'EXAMPLE-PASSWORD';
GRANT SELECT ON myDB . * TO 'dbuser'@'%';
```

   Di jendela Output, Anda melihat perintah berhasil.

Langkah 2: Buat rahasia untuk kredensi pengguna database

Selanjutnya, Anda membuat rahasia untuk menyimpan kredensil pengguna yang baru saja Anda buat, dan Anda mengaktifkan rotasi otomatis, termasuk rotasi langsung. Secrets Manager memutar rahasia, yang berarti kata sandi dibuat secara terprogram - tidak ada manusia yang melihat kata sandi baru ini. Memulai rotasi segera juga dapat membantu Anda menentukan apakah rotasi diatur dengan benar.

1. Buka konsol Secrets Manager di https://console.aws.amazon.com/secretsmanager/.
2. Pilih Simpan rahasia baru.
3. Pada halaman Pilih jenis rahasia, lakukan hal berikut:
   a. Untuk jenis Rahasia, pilih Kredensial untuk database Amazon RDS.
   b. Untuk Kredensial, masukkan nama pengguna **dbuser** dan kata sandi yang Anda masukkan untuk pengguna database yang Anda buat menggunakan MySQL Workbench.
   c. Untuk Database, pilih secretsmanagertutorialdb.
   d. Pilih Berikutnya.
4. Pada halaman Konfigurasi rahasia, untuk nama Rahasia, masukkan **SecretsManagerTutorialDbuser** dan kemudian pilih Berikutnya.
5. Pada halaman Konfigurasi rotasi, lakukan hal berikut:
   a. Nyalakan Rotasi otomatis.
   b. Untuk jadwal Rotasi, atur jadwal Hari: **2** Hari dengan Durasi:**2h**. Tetap Putar segera dipilih.
   c. Untuk fungsi Rotasi, pilih Buat fungsi rotasi, dan kemudian untuk nama fungsi, masukkan**tutorial-single-user-rotation**.

d. Untuk strategi Rotasi, pilih Single user.

e. Pilih Berikutnya.

6. Pada halaman Review, pilih Store.

   Secrets Manager kembali ke halaman detail rahasia. Di bagian atas halaman, Anda dapat melihat status konfigurasi rotasi. Secrets Manager menggunakan CloudFormation untuk membuat sumber daya seperti fungsi rotasi Lambda dan peran eksekusi yang menjalankan fungsi Lambda. Setelah CloudFormation selesai, spanduk berubah menjadi Rahasia yang dijadwalkan untuk rotasi. Rotasi pertama selesai.

Langkah 3: Uji kata sandi yang diputar

Setelah rotasi rahasia pertama, yang mungkin memakan waktu beberapa detik, Anda dapat memeriksa bahwa rahasia masih berisi kredenal yang valid. Kata sandi dalam rahasia telah berubah dari kredensi asli.

Untuk mengambil kata sandi baru dari rahasia

1. Buka konsol Secrets Manager di https://console.aws.amazon.com/secretsmanager/.
2. Pilih Rahasia, lalu pilih rahasianya**`SecretsManagerTutorialDbuser`**.
3. Pada halaman Detail rahasia, gulir ke bawah dan pilih Ambil nilai rahasia.
4. Dalam Key/valuetabel, salin nilai Rahasia untuk**`password`**.

Untuk menguji kredensialnya

1. Di MySQL Workbench, klik kanan koneksi dan kemudian pilih Edit SecretsManagerTutorialKoneksi.
2. Dalam kotak dialog Kelola Koneksi Server, untuk Nama Pengguna**`dbuser`**, masukkan, lalu pilih Tutup.
3. Kembali di MySQL Workbench, pilih koneksi. SecretsManagerTutorial
4. Dalam Buka Koneksi SSH kotak dialog, untuk Kata Sandi, tempel kata sandi yang Anda ambil dari rahasia, lalu pilih OK.

   Jika kredensialnya valid, maka MySQL Workbench terbuka ke halaman desain untuk database.

Langkah 4: Bersihkan Sumber Daya

Untuk menghindari potensi biaya, hapus rahasia yang Anda buat dalam tutorial ini. Untuk petunjuk, lihat the section called “Hapus rahasia”.

Untuk membersihkan sumber daya yang dibuat dalam tutorial sebelumnya, lihatthe section called “Langkah 4: Bersihkan Sumber Daya”.

Langkah selanjutnya

- Pelajari cara mengambil rahasia di aplikasi Anda. Lihat Dapatkan rahasia.
- Pelajari tentang jadwal rotasi lainnya. Lihat the section called “Jadwal rotasi”.

Buat AWS Secrets Manager Rahasia

Rahasia dapat berupa kata sandi, satu set kredenSIAL seperti nama pengguna dan kata sandi, token OAuth, atau informasi rahasia lainnya yang Anda simpan dalam bentuk terenkripsi di Manajer Rahasia.

> **Tip**
>
> Untuk kredentif pengguna admin Amazon RDS dan Amazon Redshift, kami sarankan Anda menggunakan rahasia terkel ola. Anda membuat rahasia terkelola melalui layanan pengelolaan, dan kemudian Anda dapat menggunakan rotasi terkelola.

Bila Anda menggunakan konsol untuk menyimpan kredentif database untuk database sumber yang direplikasi ke Wilayah lain, rahasia berisi informasi koneksi untuk database sumber. Jika Anda kemudian mereplikasi rahasia, replika adalah salinan dari rahasia sumber dan berisi informasi koneksi yang sama. Anda dapat menambahkan key/value pasangan tambahan ke rahasia untuk informasi koneksi regional.

Untuk membuat rahasia, Anda memerlukan izin yang diberikan oleh kebijakan ter SecretsManagerReadWrite kelola.

Manajer Rahasia menghasilkan entri CloudTrail log saat Anda membuat rahasia. Untuk informasi selengkapnya, lihat the section called “Masuk dengan AWS CloudTrail ”.

Untuk membuat rahasia (konsol)

1. Buka konsol Secrets Manager di https://console.aws.amazon.com/secretsmanager/.
2. Pilih Simpan rahasia baru.
3. Pada halaman Pilih jenis rahasia, lakukan hal berikut:
   a. Untuk tipe Rah asia, lakukan salah satu hal berikut:
      - Untuk menyimpan kredenSIAL database, pilih jenis kredenSIAL database yang akan disimpan. Kemudian pilih Database dan kemudian masukkan Cre dentials.
      - Untuk menyimpan kunci API, token akses, kredenSIAL yang bukan untuk database, pilih Jenis rahasia lainnya.

Secara Key/value berpasangan, masukkan rahasia Anda dalam Key/value pasangan JSON, atau pilih tab Plaintext dan masukkan rahasia dalam format apa pun. Anda dapat menyimpan hingga 65536 byte di rahasia. Beberapa contoh:

API key

Masukkan sebagai key/value pasangan:

ClientID : *my_client_id*

ClientSecret : *wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY*

OAuth token

Masukkan sebagai plaintext:

AKIAI44QH8DHBEXAMPLE

Digital certificate

Masukkan sebagai plaintext:

```
-----BEGIN CERTIFICATE-----
EXAMPLE
-----END CERTIFICATE-----
```

Private key

Masukkan sebagai plaintext:

```
---- BEGIN PRIVATE KEY ----
EXAMPLE
---- END PRIVATE KEY ----
```

- Untuk menyimpan rahasia eksternal yang dikelola dari mitra Manajer Rahasia, pilih Rahasia mitra. Kemudian pilih pasangan dan berikan detail yang mengidentifikasi rahasia untuk pasangan. Lihat perinciannya di Penggunaan AWS Secrets Manager rahasia eksternal yang dikelola untuk mengelola rahasia Pihak Ketiga.

b. Untuk kunci enkripsi, pilih AWS KMS key yang digunakan Manajer Rahasia untuk mengenkripsi nilai rahasia. Untuk informasi selengkapnya, lihat Enkripsi rahasia dan dekripsi.

- Untuk sebagian besar kasus, pilih aws/secretsmanager untuk menggunakan Manajer Rahasia. Kunci yang dikelola AWS Tidak ada biaya untuk menggunakan kunci ini.
- Jika Anda perlu mengakses rahasia dari yang lain Akun AWS, atau jika Anda ingin menggunakan kunci KMS Anda sendiri sehingga Anda dapat memutarnya atau menerapkan kebijakan kunci padanya, pilih kunci yang dikelola pelanggan dari daftar, masukkan kunci ARN atau alias ARN dari kunci yang dikelola pelanggan, atau pilih Tambahkan kunci baru untuk membuatnya. Untuk informasi tentang biaya penggunaan kunci yang dikelola pelanggan, lihatHarga.

  Anda harus punyathe section called “Izin untuk kunci KMS”. Untuk informasi tentang akses lintas akun, lihatthe section called “Cross-accountakses”.

c. Pilih Berikutnya.

4. Pada halaman Konfigurasi rahasia, lakukan hal berikut:

   a. Masukkan nama Rahasia deskriptif dan Deskripsi. Nama rahasia dapat berisi 1-512 alfanumerik dan /_+ =.@- karakter.

   b. (Opsional) Jika Anda membuat rahasia eksternal, masukkan metadata yang diperlukan oleh mitra Manajer Rahasia yang menyimpan rahasia tersebut.

   c. (Opsional) Di bagian Tag, tambahkan tag ke rahasia Anda. Untuk strategi penandaan, lihatthe section called “Kata kunci rahasia”. Jangan menyimpan informasi sensitif dalam tag karena tidak dienkripsi.

   d. (Opsional) Di Izin sumber daya, untuk menambahkan kebijakan sumber daya ke rahasia Anda, pilih Edit izin. Untuk informasi selengkapnya, lihat the section called “Resource-based kebijakan”.

   e. (Opsional) Di Rah asia replikasi, untuk mereplikasi rahasia Anda ke yang lain Wilayah AWS, pilih Rep likasi rahasia. Anda dapat mereplikasi rahasia Anda sekarang atau kembali dan mereplikasinya nanti. Untuk informasi selengkapnya, lihat Multi-region replikasi.

   f. Pilih Berikutnya.

5. (Opsional) Pada halaman Konfigurasi rotasi, Anda dapat mengaktifkan rotasi otomatis. Anda juga dapat mematikan rotasi untuk saat ini dan kemudian menyalakannya nanti. Untuk informasi selengkapnya, lihat Merotasi rahasia . Pilih Berikutnya.

6. Di halaman Tinjauan, tinjau detail rahasia Anda, lalu pilih Store.

   Manajer Rahasia kembali ke daftar rahasia. Jika rahasia baru Anda tidak muncul, pilih tombol refresh.

AWS CLI

Ketika Anda memasukkan perintah dalam shell perintah, ada risiko riwayat perintah diakses atau utilitas memiliki akses ke parameter perintah Anda. Lihat the section called “Mengurangi risiko menggunakan AWS CLI untuk menyimpan rahasia Anda AWS Secrets Manager”.

Example Buat rahasia dari kredentif database dalam file JSON

`create-secret`Contoh berikut membuat rahasia dari kredenSIAL dalam file. Untuk informasi selengkapnya, lihat AWS CLI Memuat parameter dari file di Panduan AWS CLI Pengguna.

Agar Manajer Rahasia dapat memutar rahasia, Anda harus memastikan JSON cocok dengan Struktur JSON dari sebuah rahasia.

```
aws secretsmanager create-secret \
    --name MyTestSecret \
    --secret-string file://mycreds.json
```

Isi mycreds.json:

```
{
    "engine": "mysql",
    "username": "saanvis",
    "password": "EXAMPLE-PASSWORD",
    "host": "my-database-endpoint.us-west-2.rds.amazonaws.com",
    "dbname": "myDatabase",
    "port": "3306"
}
```

Example Buat rahasia

`create-secret`Contoh berikut membuat rahasia dengan dua pasangan kunci-nilai.

```
aws secretsmanager create-secret \
    --name MyTestSecret \
    --description "My test secret created with the CLI." \
    --secret-string '{"user":"diegor","password":"EXAMPLE-PASSWORD"}'
```

Example Buat rahasia

`create-secret`Contoh berikut membuat rahasia dengan dua tag.

```
aws secretsmanager create-secret \
    --name MyTestSecret \
    --description "My test secret created with the CLI." \
    --secret-string '{"user":"diegor","password":"EXAMPLE-PASSWORD"}'  \
    --tags '[{"Key": "FirstTag", "Value": "FirstValue"}, {"Key": "SecondTag", "Value":
 "SecondValue"}]'
```

AWS SDK

Untuk membuat rahasia dengan menggunakan salah satu AWS SDK, gunakan `CreateSecret` tindakan. Lihat informasi yang lebih lengkap di the section called “AWS SDK”.

Apa yang ada dalam rahasia Secrets Manager?

Dalam Secrets Manager, rahasia terdiri dari informasi rahasia, nilai rahasia, ditambah metadata tentang rahasia. Nilai rahasia dapat berupa string atau biner.

Untuk menyimpan beberapa nilai string dalam satu rahasia, kami sarankan Anda menggunakan string teks JSON dengan pasangan nilai kunci, misalnya:

```
{
  "host"       : "ProdServer-01.databases.example.com",
  "port"       : "8888",
  "username"   : "administrator",
  "password"   : "EXAMPLE-PASSWORD",
  "dbname"     : "MyDatabase",
  "engine"     : "mysql"
}
```

Untuk rahasia database, jika Anda ingin mengaktifkan rotasi otomatis, rahasia harus berisi informasi koneksi untuk database dalam struktur JSON yang benar. Untuk informasi selengkapnya, lihat the section called “Struktur JSON dari sebuah rahasia”.

Metadata

Metadata rahasia meliputi:

- Nama Sumber Daya Amazon (ARN) dengan format berikut:

```
arn:aws:secretsmanager:<Region>:<AccountId>:secret:SecretName-6RandomCharacters
```

Secrets Manager mencakup enam karakter acak di akhir nama rahasia untuk membantu memastikan bahwa ARN rahasia itu unik. Jika rahasia asli dihapus, dan kemudian rahasia baru dibuat dengan nama yang sama, kedua rahasia memiliki ARN yang berbeda karena karakter ini. Pengguna dengan akses ke rahasia lama tidak secara otomatis mendapatkan akses ke rahasia baru karena ARN berbeda.

- Nama rahasia, deskripsi, kebijakan sumber daya, dan tag.
- ARN untuk kunci enkripsi, yang digunakan Secrets Manager untuk mengenkripsi dan mendekripsi nilai rahasia. AWS KMS key Secrets Manager menyimpan teks rahasia dalam bentuk terenkripsi dan mengenkripsi rahasia dalam perjalanan. Lihat the section called “Enkripsi rahasia dan dekripsi”.
- Informasi tentang cara memutar rahasia, jika Anda mengatur rotasi. Lihat Merotasi rahasia .

Secrets Manager menggunakan kebijakan izin IAM untuk memastikan bahwa hanya pengguna yang berwenang yang dapat mengakses atau memodifikasi rahasia. Lihat Kendali autentikasi dan akses untuk AWS Secrets Manager.

Rahasia memiliki versi yang menyimpan salinan nilai rahasia terenkripsi. Ketika Anda mengubah nilai rahasia, atau ketika rahasia dirotasi, Secrets Manager membuat versi baru. Lihat the section called “Versi rahasia”.

Anda dapat menggunakan rahasia di beberapa Wilayah AWS dengan mereplikasi itu. Ketika Anda mereplikasi rahasia, Anda membuat salinan rahasia asli atau primer yang disebut rahasia replika. Rahasia replika tetap terkait dengan rahasia utama. Lihat Multi-region replikasi.

Lihat Kelola rahasia.

Versi rahasia

Rahasia memiliki versi yang menyimpan salinan nilai rahasia terenkripsi. Ketika Anda mengubah nilai rahasia, atau ketika rahasia dirotasi, Secrets Manager membuat versi baru.

Secrets Manager tidak menyimpan riwayat linear rahasia dengan versi. Sebagai gantinya, ia melacak tiga versi tertentu dengan memberi label:

- Versi saat ini - `AWSCURRENT`

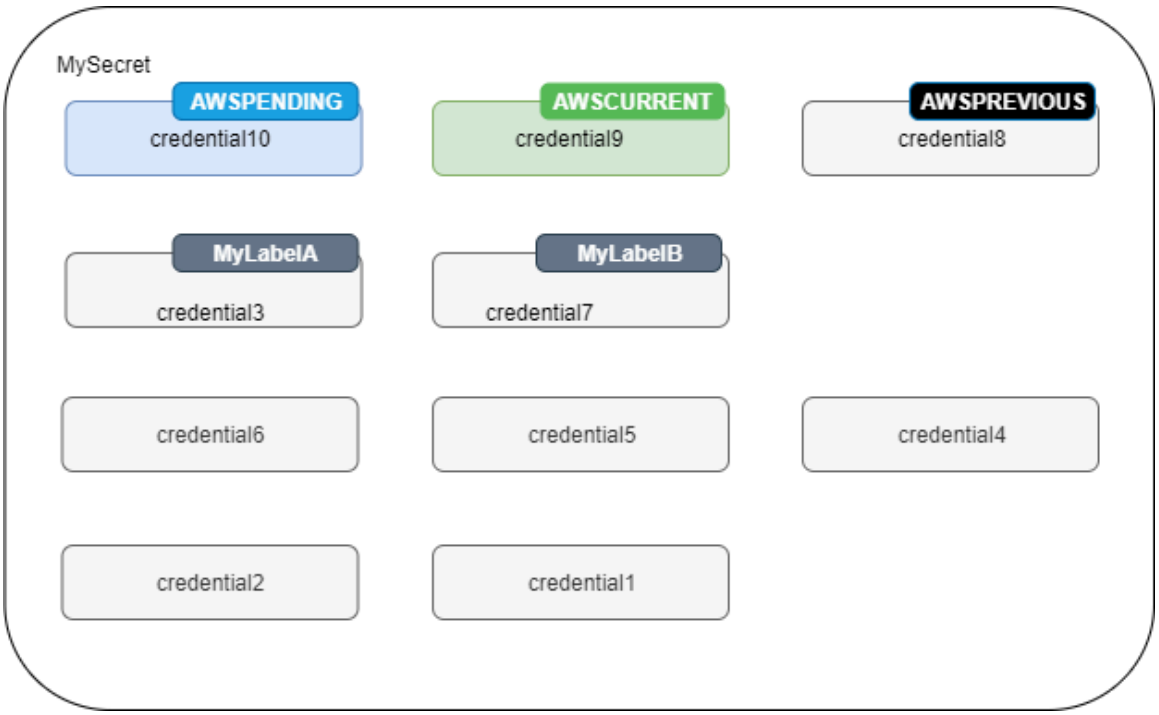
- Versi sebelumnya - `AWSPREVIOUS`
- Versi yang tertunda (selama rotasi) — `AWSPENDING`

Rahasia selalu memiliki versi berlabel `AWSCURRENT`, dan Secrets Manager mengembalikan versi tersebut secara default ketika Anda mengambil nilai rahasia.

Anda juga dapat melabeli versi dengan label Anda sendiri `update-secret-version-stage`dengan memanggil AWS CLI. Anda dapat melampirkan hingga 20 label ke versi secara rahasia. Dua versi rahasia tidak dapat memiliki label pementasan yang sama. Versi dapat memiliki beberapa label.

Secrets Manager tidak pernah menghapus versi berlabel, tetapi versi yang tidak berlabel dianggap tidak digunakan lagi. Secrets Manager menghapus versi yang tidak digunakan lagi jika ada lebih dari 100. Secrets Manager tidak menghapus versi yang dibuat kurang dari 24 jam yang lalu.

Gambar berikut menunjukkan rahasia yang memiliki versi AWS berlabel dan versi berlabel pelanggan. Versi tanpa label dianggap usang dan akan dihapus oleh Secrets Manager di beberapa titik di masa mendatang.



Struktur rahasia JSON AWS Secrets Manager

Anda dapat menyimpan teks atau biner apa pun dalam rahasia Secrets Manager hingga ukuran maksimum 65.536 Byte.

Jika Anda menggunakanthe section called “Rotasi dengan fungsi Lambda”, rahasia harus berisi bidang JSON tertentu yang diharapkan oleh fungsi rotasi. Misalnya, untuk rahasia yang berisi kredensil database, fungsi rotasi terhubung ke database untuk memperbarui kredensil, sehingga rahasia harus berisi informasi koneksi database.

Jika Anda menggunakan konsol untuk mengedit rotasi rahasia database, rahasia harus berisi pasangan nilai kunci JSON tertentu yang mengidentifikasi database. Secrets Manager menggunakan bidang ini untuk menanyakan database untuk menemukan VPC yang benar untuk menyimpan fungsi rotasi.

Nama kunci JSON peka huruf besar/kecil.

Topik

- Kredensi Amazon RDS dan Aurora
- Kredensi Amazon Redshift
- Kredensi Tanpa Server Amazon Redshift
- Kredensi Amazon DocumentDB
- Amazon Timestream untuk struktur rahasia InfluxDB
- ElastiCache Kredensi Amazon
- Kredensil Direktori Aktif

Kredensi Amazon RDS dan Aurora

Untuk menggunakan template fungsi rotasi yang disediakan oleh Secrets Manager, gunakan struktur JSON berikut. Anda dapat menambahkan lebih banyak key/value pasangan, misalnya untuk memuat informasi koneksi untuk database replika di Wilayah lain.

DB2

> Untuk instans Amazon RDS Db2, karena pengguna tidak dapat mengubah kata sandi mereka sendiri, Anda harus memberikan kredensi admin dalam rahasia terpisah.

```
{
```

```
  "engine": "db2",
  "host": "<instance host name/resolvable DNS name>",
  "username": "<username>",
  "password": "<password>",
  "dbname": "<database name. If not specified, defaults to None>",
  "port": <TCP port number. If not specified, defaults to 3306>,
  "masterarn": "<ARN of the elevated secret>",
  "dbInstanceIdentifier": <optional: ID of the instance. Alternately, use
 dbClusterIdentifier. Required for configuring rotation in the console.>",
  "dbClusterIdentifier": <optional: ID of the cluster. Alternately, use
 dbInstanceIdentifier. Required for configuring rotation in the console.>"
}
```

MariaDB

```
{
  "engine": "mariadb",
  "host": "<instance host name/resolvable DNS name>",
  "username": "<username>",
  "password": "<password>",
  "dbname": "<database name. If not specified, defaults to None>",
  "port": <TCP port number. If not specified, defaults to 3306>,
  "masterarn": "<optional: ARN of the elevated secret. Required for the the section
 called “Pengguna bergantian”.>",
  "dbInstanceIdentifier": <optional: ID of the instance. Alternately, use
 dbClusterIdentifier. Required for configuring rotation in the console.>",
  "dbClusterIdentifier": <optional: ID of the cluster. Alternately, use
 dbInstanceIdentifier. Required for configuring rotation in the console.>"
}
```

MySQL

```
{
  "engine": "mysql",
  "host": "<instance host name/resolvable DNS name>",
  "username": "<username>",
  "password": "<password>",
  "dbname": "<database name. If not specified, defaults to None>",
  "port": <TCP port number. If not specified, defaults to 3306>,
  "masterarn": "<optional: ARN of the elevated secret. Required for the the section
 called “Pengguna bergantian”.>",
  "dbInstanceIdentifier": <optional: ID of the instance. Alternately, use
 dbClusterIdentifier. Required for configuring rotation in the console.>",
```

```
  "dbClusterIdentifier": <optional: ID of the cluster. Alternately, use
 dbInstanceIdentifier. Required for configuring rotation in the console.>"
}
```

Oracle

```
{
  "engine": "oracle",
  "host": "<instance host name/resolvable DNS name>",
  "username": "<username>",
  "password": "<password>",
  "dbname": "<database name>",
  "port": <TCP port number. If not specified, defaults to 1521>,
  "masterarn": "<optional: ARN of the elevated secret. Required for the the section
 called “Pengguna bergantian”.>",
  "dbInstanceIdentifier": <optional: ID of the instance. Alternately, use
 dbClusterIdentifier. Required for configuring rotation in the console.>",
  "dbClusterIdentifier": <optional: ID of the cluster. Alternately, use
 dbInstanceIdentifier. Required for configuring rotation in the console.>"
}
```

Postgres

```
{
  "engine": "postgres",
  "host": "<instance host name/resolvable DNS name>",
  "username": "<username>",
  "password": "<password>",
  "dbname": "<database name. If not specified, defaults to 'postgres'>",
  "port": <TCP port number. If not specified, defaults to 5432>,
  "masterarn": "<optional: ARN of the elevated secret. Required for the the section
 called “Pengguna bergantian”.>",
  "dbInstanceIdentifier": <optional: ID of the instance. Alternately, use
 dbClusterIdentifier. Required for configuring rotation in the console.>",
  "dbClusterIdentifier": <optional: ID of the cluster. Alternately, use
 dbInstanceIdentifier. Required for configuring rotation in the console.>"
}
```

SQLServer

```
{
  "engine": "sqlserver",
  "host": "<instance host name/resolvable DNS name>",
```

```
  "username": "<username>",
  "password": "<password>",
  "dbname": "<database name. If not specified, defaults to 'master'>",
  "port": <TCP port number. If not specified, defaults to 1433>,
  "masterarn": "<optional: ARN of the elevated secret. Required for the the section
 called “Pengguna bergantian”.>",
  "dbInstanceIdentifier": <optional: ID of the instance. Alternately, use
 dbClusterIdentifier. Required for configuring rotation in the console.>",
  "dbClusterIdentifier": <optional: ID of the cluster.Alternately, use
 dbInstanceIdentifier.  Required for configuring rotation in the console.>"
}
```

Kredensi Amazon Redshift

Untuk menggunakan template fungsi rotasi yang disediakan oleh Secrets Manager, gunakan struktur JSON berikut. Anda dapat menambahkan lebih banyak key/value pasangan, misalnya untuk memuat informasi koneksi untuk database replika di Wilayah lain.

```
{
  "engine": "redshift",
  "host": "<instance host name/resolvable DNS name>",
  "username": "<username>",
  "password": "<password>",
  "dbname": "<database name. If not specified, defaults to None>",
  "dbClusterIdentifier": "<optional: database ID. Required for configuring rotation in
 the console.>"
  "port": <optional: TCP port number. If not specified, defaults to 5439>
  "masterarn": "<optional: ARN of the elevated secret. Required for the the section
 called “Pengguna bergantian”.>"
}
```

Kredensi Tanpa Server Amazon Redshift

Untuk menggunakan template fungsi rotasi yang disediakan oleh Secrets Manager, gunakan struktur JSON berikut. Anda dapat menambahkan lebih banyak key/value pasangan, misalnya untuk memuat informasi koneksi untuk database replika di Wilayah lain.

```
{
  "engine": "redshift",
  "host": "<instance host name/resolvable DNS name>",
  "username": "<username>",
```

```
  "password": "<password>",
  "dbname": "<database name. If not specified, defaults to None>",
  "namespaceName": "<optional: namespace name, Required for configuring rotation in the
 console.> "
  "port": <optional: TCP port number. If not specified, defaults to 5439>
  "masterarn": "<optional: ARN of the elevated secret. Required for the the section
 called “Pengguna bergantian”.>"
}
```

Kredensi Amazon DocumentDB

Untuk menggunakan template fungsi rotasi yang disediakan oleh Secrets Manager, gunakan struktur JSON berikut. Anda dapat menambahkan lebih banyak key/value pasangan, misalnya untuk memuat informasi koneksi untuk database replika di Wilayah lain.

```
{
  "engine": "mongo",
  "host": "<instance host name/resolvable DNS name>",
  "username": "<username>",
  "password": "<password>",
  "dbname": "<database name. If not specified, defaults to None>",
  "port": <TCP port number. If not specified, defaults to 27017>,
  "ssl": <true|false. If not specified, defaults to false>,
  "masterarn": "<optional: ARN of the elevated secret. Required for the the section
 called “Pengguna bergantian”.>",
  "dbClusterIdentifier": "<optional: database cluster ID. Alternately, use
 dbInstanceIdentifier. Required for configuring rotation in the console.>"
  "dbInstanceIdentifier": "<optional: database instance ID. Alternately, use
 dbClusterIdentifier. Required for configuring rotation in the console.>"
}
```

Amazon Timestream untuk struktur rahasia InfluxDB

Untuk memutar rahasia Timestream, Anda dapat menggunakan template the section called “Amazon Timestream untuk InfluxDB” rotasi.

Untuk informasi selengkapnya, lihat Cara Amazon TimeStream untuk InfluxDB menggunakan rahasia di Panduan Pengembang Amazon Timestream.

Rahasia Timestream harus dalam struktur JSON yang benar untuk dapat menggunakan template rotasi. Untuk informasi selengkapnya, lihat Apa yang ada di rahasia di Panduan Pengembang Amazon Timestream.

ElastiCache Kredensi Amazon

Contoh berikut menunjukkan struktur JSON untuk rahasia yang menyimpan ElastiCache kredensil.

```
{
  "password": "<password>",
  "username": "<username>"
  "user_arn": "ARN of the Amazon EC2 user"
}
```

Untuk informasi selengkapnya, lihat Memutar kata sandi secara otomatis untuk pengguna di Panduan ElastiCache Pengguna Amazon.

Kredensil Direktori Aktif

AWS Directory Service menggunakan rahasia untuk menyimpan kredensil Active Directory. Untuk informasi selengkapnya, lihat Menggabungkan instans Amazon EC2 Linux dengan mulus ke Direktori Aktif AD Terkelola di Panduan AWS Directory Service Administrasi. Gabungan domain yang mulus memerlukan nama kunci dalam contoh berikut. Jika Anda tidak menggunakan gabungan domain tanpa batas, Anda dapat mengubah nama kunci dalam rahasia menggunakan variabel lingkungan seperti yang dijelaskan dalam kode templat fungsi rotasi.

Untuk memutar rahasia Active Directory, Anda dapat menggunakan template rotasi Active Directory.

Active Directory credential

```
{
  "awsSeamlessDomainUsername": "<username>",
  "awsSeamlessDomainPassword": "<password>"
}
```

Jika Anda ingin memutar rahasia, Anda menyertakan ID direktori domain.

```
{
  "awsSeamlessDomainDirectoryId": "d-12345abc6e",
  "awsSeamlessDomainUsername": "<username>",
  "awsSeamlessDomainPassword": "<password>"
}
```

Jika rahasia digunakan bersama dengan rahasia yang berisi keytab, Anda menyertakan ARN rahasia keytab.

```
{
  "awsSeamlessDomainDirectoryId": "d-12345abc6e",
  "awsSeamlessDomainUsername": "<username>",
  "awsSeamlessDomainPassword": "<password>",
  "directoryServiceSecretVersion": 1,
  "schemaVersion": "1.0",
  "keytabArns": [
    "<ARN of child keytab secret 1>,
    "<ARN of child keytab secret 2>,
    "<ARN of child keytab secret 3>,
  ],
  "lastModifiedDateTime": "2021-07-19 17:06:58"
}
```

Active Directory keytab

Untuk informasi tentang menggunakan file tab tombol untuk mengautentikasi ke akun Active Directory di Amazon EC2, lihat Menyebarkan dan mengonfigurasi otentikasi Direktori Aktif dengan SQL Server 2017 di Amazon Linux 2.

```
{
  "awsSeamlessDomainDirectoryId": "d-12345abc6e",
  "schemaVersion": "1.0",
  "name": "< name>",
  "principals": [
    "aduser@MY.EXAMPLE.COM",
    "MSSQLSvc/test:1433@MY.EXAMPLE.COM"
  ],
  "keytabContents": "<keytab>",
  "parentSecretArn": "<ARN of parent secret>",
  "lastModifiedDateTime": "2021-07-19 17:06:58"
  "version": 1
}
```

Mengelola rahasia dengan AWS Secrets Manager

Topik

- Memperbarui nilai untuk AWS Secrets Manager Rahasia
- Buat kata sandi dengan Secrets Manager
- Kembalikan rahasia ke versi sebelumnya
- Mengubah kunci enkripsi untuk AWS Secrets Manager Rahasia
- Ubah sebuah AWS Secrets Manager Rahasia
- Temukan rahasia di AWS Secrets Manager
- Hapus sebuah AWS Secrets Manager Rahasia
- Kembalikan AWS Secrets Manager Rahasia
- Menandai rahasia di AWS Secrets Manager

Memperbarui nilai untuk AWS Secrets Manager Rahasia

Untuk memperbarui nilai rahasia Anda, Anda dapat menggunakan konsol, CLI, atau SDK. Saat Anda memperbarui nilai rahasia, Manajer Rahasia membuat versi baru rahasia dengan label `AWSCURRENT` pementasan. Anda masih dapat mengakses versi lama, yang memiliki label`AWSPREVIOUS`. Anda juga dapat menambahkan label Anda sendiri. Untuk informasi selengkapnya, lihat pembuatan versi Manajer Rahasia.

Untuk memperbarui nilai rahasia (konsol)

1. Buka konsol Secrets Manager di https://console.aws.amazon.com/secretsmanager/.
2. Dari daftar rahasia, pilih rahasia Anda.
3. Pada halaman detail rahasia, pada tab I khtisar, di bagian Nilai rahasia, pilih Ambil nilai rahasia dan kemudian pilih Edit.

AWS CLI

Untuk memperbarui nilai rahasia (AWS CLI)

- Ketika Anda memasukkan perintah dalam shell perintah, ada risiko riwayat perintah diakses atau utilitas memiliki akses ke parameter perintah Anda. Lihat the section called “Mengurangi risiko menggunakan AWS CLI untuk menyimpan rahasia Anda AWS Secrets Manager”.

  Berikut ini `put-secret-value` menciptakan versi baru rahasia dengan dua pasangan kunci-nilai.

  ```
  aws secretsmanager put-secret-value \
        --secret-id MyTestSecret \
        --secret-string "{\"user\":\"diegor\",\"password\":\"EXAMPLE-PASSWORD\"}"
  ```

  Berikut ini `put-secret-value` membuat versi baru dengan label pementasan kustom. Versi baru akan memiliki label `MyLabel` dan`AWSCURRENT`.

  ```
  aws secretsmanager put-secret-value \
        --secret-id MyTestSecret \
        --secret-string "{\"user\":\"diegor\",\"password\":\"EXAMPLE-PASSWORD\"}"
        --version-stages "MyLabel"
  ```

AWS SDK

Kami menyarankan Anda menghindari menelep `UpdateSecret` on `PutSecretValue` atau dengan kecepatan berkelanjutan lebih dari sekali setiap 10 menit. Saat Anda menelep `UpdateSecret` on `PutSecretValue` atau memperbarui nilai rahasia, Manajer Rahasia membuat versi baru rahasia. Secrets Manager menghapus versi yang tidak berlabel ketika ada lebih dari 100, tetapi tidak menghapus versi yang dibuat kurang dari 24 jam yang lalu. Jika Anda memperbarui nilai rahasia lebih dari sekali setiap 10 menit, Anda membuat lebih banyak versi daripada yang dihapus oleh Secrets Manager, dan Anda akan mencapai kuota untuk versi rahasia.

Untuk memperbarui nilai rahasia, gunakan tindakan berikut: `UpdateSecret` atau `PutSecretValue`. Lihat informasi yang lebih lengkap di the section called “AWS SDK”.

Buat kata sandi dengan Secrets Manager

Pola umum untuk menggunakan Secrets Manager adalah membuat kata sandi di Secrets Manager dan kemudian menggunakan kata sandi itu di database atau layanan Anda. Anda dapat melakukan ini menggunakan metode berikut:

- CloudFormation — LihatCloudFormation.
- AWS CLI — Lihat `get-random-password`.
- AWS SDK — Lihat `GetRandomPassword`.

Kembalikan rahasia ke versi sebelumnya

Anda dapat mengembalikan rahasia ke versi sebelumnya dengan memindahkan label yang dilampirkan ke versi rahasia menggunakan. AWS CLI Untuk informasi tentang cara Secrets Manager menyimpan versi rahasia, lihatthe section called “Versi rahasia”.

`update-secret-version-stage`Contoh berikut memindahkan label pementasan AWSCRENT ke versi rahasia sebelumnya, yang mengembalikan rahasia ke versi sebelumnya. Untuk menemukan ID untuk versi sebelumnya, gunakan `list-secret-version-ids`atau lihat versi di konsol Secrets Manager.

Untuk contoh ini, versi dengan label AWSCRENT adalah A1B2C3D4-5678-90AB-CDEF-Example11111 dan versi dengan label AWSPREVIOUS adalah A1B2C3D4-5678-90AB-CDEF-Example22222. Dalam contoh ini, Anda memindahkan label AWSCRENT dari versi 11111 ke 22222. Karena label AWSCRENT dihapus dari versi, `update-secret-version-stage` secara otomatis memindahkan label AWSPREVIOUS ke versi tersebut (11111). Efeknya adalah versi AWSCRENT dan AWSPREVIOUS ditukar.

```
aws secretsmanager update-secret-version-stage \
  --secret-id MyTestSecret \
  --version-stage AWSCURRENT \
  --move-to-version-id a1b2c3d4-5678-90ab-cdef-EXAMPLE22222 \
  --remove-from-version-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

Mengubah kunci enkripsi untuk AWS Secrets Manager Rahasia

Secrets Manager menggunakan enkripsi amplop dengan AWS KMS kunci dan kunci data untuk melindungi setiap nilai rahasia. Untuk setiap rahasia, Anda dapat memilih kunci KMS mana yang

akan digunakan. Anda dapat menggunakan Kunci yang dikelola AWS aws/secretsmanager, atau Anda dapat menggunakan kunci yang dikelola pelanggan. Untuk kebanyakan kasus, kami sarankan menggunakanaws/secretsmanager, dan tidak ada biaya untuk menggunakannya. Jika Anda perlu mengakses rahasia dari yang lain Akun AWS, atau jika Anda ingin menggunakan kunci KMS Anda sendiri sehingga Anda dapat memutarnya atau menerapkan kebijakan kunci padanya, gunakan a kunci yang dikelola pelanggan. Anda harus punyathe section called “Izin untuk kunci KMS”. Untuk informasi tentang biaya penggunaan kunci yang dikelola pelanggan, lihatHarga.

Anda dapat mengubah kunci enkripsi untuk rahasia Anda. Misalnya, jika Anda ingin mengakses rahasia dari akun lain, dan rahasia tersebut saat ini dienkripsi menggunakan kunci AWS terkelolaaws/secretsmanager, Anda dapat beralih ke kunci yang dikelola pelanggan.

> **Tip**
>
> Jika Anda ingin memutar kunci yang dikelola pelanggan, sebaiknya gunakan rotasi tombol AWS KMS otomatis. Untuk informasi selengkapnya, lihat Mem ut AWS KMS ar tombol.

Saat Anda mengubah kunci enkripsi, Secrets Manager mengenkripsi ulangAWSCURRENT,AWSPENDING, dan AWSPREVIOUS versi dengan kunci baru. Untuk menghindari mengunci Anda keluar dari rahasia, Secrets Manager menyimpan semua versi yang ada dienkripsi dengan kunci sebelumnya. Itu berarti Anda dapat mendekripsi AWSCURRENTAWSPENDING,, dan AWSPREVIOUS versi dengan kunci sebelumnya atau kunci baru. Jika Anda tidak memiliki kms:Decrypt izin ke kunci sebelumnya, saat Anda mengubah kunci enkripsi, Manajer Rahasia tidak dapat mendekripsi versi rahasia untuk mengenkripsi ulang. Dalam hal ini, versi yang ada tidak dienkripsi ulang.

Untuk membuatnya hanya AWSCURRENT dapat didekripsi oleh kunci enkripsi baru, buat versi baru rahasia dengan kunci baru. Kemudian untuk dapat mendekripsi versi AWSCURRENT rahasia, Anda harus memiliki izin ke kunci baru.

Jika Anda menonaktifkan kunci enkripsi sebelumnya, Anda tidak akan dapat mendekripsi versi rahasia apa pun kecualiAWSCURRENT,AWSPENDING, danAWSPREVIOUS. Jika Anda memiliki versi rahasia berlabel lain yang ingin Anda pertahankan akses, Anda perlu membuat ulang versi tersebut dengan kunci enkripsi baru menggunakanthe section called “AWS CLI”.

Untuk mengubah kunci enkripsi untuk rahasia (konsol)

1. Buka konsol Secrets Manager di https://console.aws.amazon.com/secretsmanager/.

2. Dari daftar rahasia, pilih rahasia Anda.
3. Pada halaman detail rahasia, di bagian Detail Rahasia, pilih Tind akan, lalu pilih Edit kunci enkripsi.

AWS CLI

Jika Anda mengubah kunci enkripsi untuk rahasia dan kemudian menonaktifkan kunci enkripsi sebelumnya, Anda tidak akan dapat mendekripsi versi rahasia apa pun kecualiAWSCURRENT,AWSPENDING, danAWSPREVIOUS. Jika Anda memiliki versi rahasia berlabel lain yang ingin Anda pertahankan akses, Anda perlu membuat ulang versi tersebut dengan kunci enkripsi baru menggunakanthe section called “AWS CLI”.

Untuk mengubah kunci enkripsi untuk rahasia (AWS CLI)

1. `update-secret`Contoh berikut memperbarui kunci KMS yang digunakan untuk mengenkripsi nilai rahasia. Kunci KMS harus berada di wilayah yang sama dengan rahasia.

```
aws secretsmanager update-secret \
      --secret-id MyTestSecret \
      --kms-key-id arn:aws:kms:us-west-2:123456789012:key/EXAMPLE1-90ab-cdef-fedc-
ba987EXAMPLE
```

2. (Opsional) Jika Anda memiliki versi rahasia yang memiliki label khusus, untuk mengenkripsi ulang menggunakan kunci baru, Anda harus membuat ulang versi tersebut.

   Ketika Anda memasukkan perintah dalam shell perintah, ada risiko riwayat perintah diakses atau utilitas memiliki akses ke parameter perintah Anda. Lihat the section called “Mengurangi risiko menggunakan AWS CLI untuk menyimpan rahasia Anda AWS Secrets Manager”.

   a. Dapatkan nilai versi rahasia.

```
aws secretsmanager get-secret-value \
      --secret-id MyTestSecret \
      --version-stage MyCustomLabel
```

      Catat nilai rahasianya.

   b. Buat versi baru dengan nilai itu.

```
aws secretsmanager put-secret-value \
```

```
    --secret-id testDescriptionUpdate \
    --secret-string "SecretValue" \
    --version-stages "MyCustomLabel"
```

Ubah sebuah AWS Secrets Manager Rahasia

Anda dapat memodifikasi metadata rahasia setelah dibuat, tergantung pada siapa yang membuat rahasia. Untuk rahasia yang dibuat oleh layanan lain, Anda mungkin perlu menggunakan layanan lain untuk memperbarui atau memutarnya.

Untuk menentukan siapa yang mengelola rahasia, Anda dapat meninjau nama rahasia. Rahasia yang dikelola oleh layanan lain diawali dengan ID layanan itu. Atau, di AWS CLI, panggil describe-secret, lalu tinjau bidang tersebut. `OwningService` Untuk informasi selengkapnya, lihat Rahasia dikelola oleh layanan lain.

Untuk rahasia yang Anda kelola, Anda dapat mengubah deskripsi, kebijakan berbasis sumber daya, kunci enkripsi, dan tag. Anda juga dapat mengubah nilai rahasia terenkripsi; Namun, kami sarankan Anda menggunakan rotasi untuk memperbarui nilai rahasia yang berisi kredenSIAL. Rotasi memperbarui rahasia di Secrets Manager dan kredenSIAL pada database atau layanan. Ini membuat rahasia disinkronkan secara otomatis sehingga ketika klien meminta nilai rahasia, mereka selalu mendapatkan satu set kredenSIAL yang berfungsi. Untuk informasi selengkapnya, lihat Merotasi rahasia .

Manajer Rahasia menghasilkan entri CloudTrail log saat Anda memodifikasi rahasia. Untuk informasi selengkapnya, lihat the section called “Masuk dengan AWS CloudTrail ”.

Untuk memperbarui rahasia yang Anda kelola (konsol)

1. Buka konsol Secrets Manager di https://console.aws.amazon.com/secretsmanager/.
2. Dari daftar rahasia, pilih rahasia Anda.
3. Pada halaman detail rahasia, lakukan salah satu hal berikut:

   Perhatikan bahwa Anda tidak dapat mengubah nama atau ARN rahasia.

   - Untuk memperbarui deskripsi, di bagian detail Rahasia, pilih Tind akan, lalu pilih Edit deskripsi.
   - Untuk memperbarui kunci enkripsi, lihatthe section called “Mengubah kunci enkripsi untuk rahasia”.

- Untuk memperbarui tag, pada tab Tag, pilih Edit tag. Lihat the section called “Kata kunci rahasia”.
- Untuk memperbarui nilai rahasia, lihatthe section called “Memperbarui nilai rahasia”.
- Untuk memperbarui izin untuk rahasia Anda, pada tab Ikh tisar, pilih Edit izin. Lihat the section called “Resource-based kebijakan”.

- Untuk memperbarui rotasi untuk rahasia Anda, pada tab Rotasi, pilih Edit rotasi. Lihat Merotasi rahasia .
- Untuk mereplikasi rahasia Anda ke Wilayah lain, lihatMulti-region replikasi.
- Jika rahasia Anda memiliki replika, Anda dapat mengubah kunci enkripsi untuk replika. Pada tab Replikasi, pilih tombol radio untuk replika, lalu pada menu Tind akan, pilih Edit kunci enkripsi. Lihat the section called “Enkripsi rahasia dan dekripsi”.
- Untuk mengubah rahasia sehingga dikelola oleh layanan lain, Anda perlu membuat ulang rahasia di layanan itu. Lihat Rahasia dikelola oleh layanan lain.

AWS CLI

Example Perbarui deskripsi rahasia

`update-secret`Contoh berikut memperbarui deskripsi rahasia.

```
aws secretsmanager update-secret \
    --secret-id MyTestSecret \
    --description "This is a new description for the secret."
```

AWS SDK

Kami menyarankan Anda menghindari menelep `UpdateSecret` on `PutSecretValue` atau dengan kecepatan berkelanjutan lebih dari sekali setiap 10 menit. Saat Anda menelep `UpdateSecret` on `PutSecretValue` atau memperbarui nilai rahasia, Manajer Rahasia membuat versi baru rahasia. Secrets Manager menghapus versi yang tidak berlabel ketika ada lebih dari 100, tetapi tidak menghapus versi yang dibuat kurang dari 24 jam yang lalu. Jika Anda memperbarui nilai rahasia lebih dari sekali setiap 10 menit, Anda membuat lebih banyak versi daripada yang dihapus oleh Secrets Manager, dan Anda akan mencapai kuota untuk versi rahasia.

Untuk memperbarui rahasia, gunakan tindakan berikut: `UpdateSecret` atau `ReplicateSecretToRegions`. Lihat informasi yang lebih lengkap di the section called “AWS SDK”.

Temukan rahasia di AWS Secrets Manager

Saat Anda mencari rahasia tanpa filter, Manajer Rahasia mencocokkan kata kunci dalam nama rahasia, deskripsi, kunci tag, dan nilai tag. Pencarian tanpa filter tidak peka huruf besar/kecil dan mengabaikan karakter khusus, seperti spasi,/, _, =, #, dan hanya menggunakan angka dan huruf. Saat Anda mencari tanpa filter, Manajer Rahasia menganalisis string pencarian untuk mengubahnya menjadi kata-kata terpisah. Kata-kata dipisahkan oleh perubahan apa pun dari huruf besar ke huruf kecil, dari huruf ke angka, atau dari number/letter ke tanda baca. Misalnya, memasukkan istilah `credsDatabase#892` pencarian mencari`creds`,`Database`, dan `892` dalam nama, deskripsi, dan kunci dan nilai tag.

Manajer Rahasia menghasilkan entri CloudTrail log ketika Anda mencantumkan rahasia. Untuk informasi selengkapnya, lihat the section called “Masuk dengan AWS CloudTrail ”.

Manajer Rahasia adalah layanan regional dan hanya rahasia di wilayah yang dipilih yang dikembalikan.

Penelusuran filter

Jika Anda tidak menggunakan filter apa pun, Manajer Rahasia memecah string pencarian menjadi kata-kata dan kemudian mencari semua atribut untuk kecocokan. Pencarian ini tidak peka huruf besar/kecil. Misalnya, mencari rahasia **`My_Secret`** yang cocok dengan kata `my` atau `secret` dalam nama, deskripsi, atau tag.

Anda dapat menerapkan filter berikut untuk pencarian Anda:

Nama

Cocokkan dengan awal nama rahasia; peka huruf besar/kecil. Misalnya, `Nama:` **`Data`** mengembalikan rahasia bernama`DatabaseSecret`, tetapi tidak `databaseSecret` atau`MyData`.

Deskripsi

Cocokkan kata-kata dalam deskripsi rahasia, tidak peka huruf besar-kecil. Misalnya, `Deskripsi:` men **`My Description`** cocokkan rahasia dengan deskripsi berikut:

- `My Description`
- `my description`
- `My basic description`
- `Description of my secret`

Dikelola oleh

Menemukan rahasia yang dikelola oleh layanan di luar AWS, misalnya:

- 1Kata Sandi
- Akeyless
- CyberArk
- HashiCorp

Memiliki layanan

Cocokkan dengan awal awalan ID layanan pengelolaan, tidak peka huruf besar-kecil. Misalnya, men **`my-ser`** cocokkan rahasia yang dikelola oleh layanan dengan awalan `my-serv` dan`my-service`. Untuk informasi selengkapnya, lihat Rahasia dikelola oleh layanan lain.

Rahasia yang direplikasi

Anda dapat memfilter rahasia utama, rahasia replika, atau rahasia yang tidak direplikasi.

Kunci tag

Cocokkan dengan awal kunci tag; peka huruf besar/kecil. Misalnya, kunci Tag: **`Prod`** mengembalikan rahasia dengan tag `Production` dan`Prod1`, tetapi bukan rahasia dengan tag `prod` atau`1 Prod`.

Nilai tag

Cocokkan dengan awal nilai tag; peka huruf besar/kecil. Misalnya, nilai tag: **`Prod`** mengembalikan rahasia dengan tag `Production` dan`Prod1`, tetapi bukan rahasia dengan nilai tag `prod` atau`1 Prod`.

AWS CLI

Example Buat daftar rahasia di akun Anda

`list-secrets`Contoh berikut mendapatkan daftar rahasia di akun Anda.

```
aws secretsmanager list-secrets
```

Example Filter daftar rahasia di akun Anda

`list-secrets`Contoh berikut mendapatkan daftar rahasia di akun Anda yang memiliki Test dalam nama. Pemfilteran berdasarkan nama peka huruf besar/kecil.

```
aws secretsmanager list-secrets \
    --filters Key="name",Values="Test"
```

Example Temukan rahasia yang dikelola oleh orang lain AWS layanan

`list-secrets`Contoh berikut mendapatkan daftar rahasia yang dikelola oleh layanan. Anda menentukan layanan dengan ID. Untuk informasi selengkapnya, lihat Rahasia dikelola oleh layanan lain.

```
aws secretsmanager list-secrets \
    --filters Key="owning-service",Values="<service ID prefix>"
```

AWS SDK

Untuk menemukan rahasia dengan menggunakan salah satu AWS SDK, gunakan `ListSecrets`. Lihat informasi yang lebih lengkap di the section called “AWS SDK”.

Hapus sebuah AWS Secrets Manager Rahasia

Karena sifat kritis rahasia, AWS Secrets Manager sengaja membuat penghapusan rahasia menjadi sulit. Manajer Rahasia tidak segera menghapus rahasia. Sebaliknya, Secrets Manager segera membuat rahasia tidak dapat diakses dan dijadwalkan untuk dihapus setelah jendela pemulihan minimal tujuh hari. Sampai jendela pemulihan berakhir, Anda dapat memulihkan rahasia yang sebelumnya Anda hapus. Tidak ada biaya untuk rahasia yang telah Anda tandai untuk dihapus.

Anda tidak dapat menghapus rahasia utama jika direplikasi ke Wilayah lain. Pertama hapus replika, lalu hapus rahasia utama. Ketika Anda menghapus replika, replika akan dihapus segera.

Anda tidak dapat langsung menghapus versi rahasia. Sebagai gantinya, Anda menghapus semua label pementasan dari versi menggunakan AWS SDK AWS CLI atau. Ini menandai versi sebagai usang, dan kemudian Manajer Rahasia dapat secara otomatis menghapus versi di latar belakang.

Jika Anda tidak tahu apakah aplikasi masih menggunakan rahasia, Anda dapat membuat CloudWatch alarm Amazon untuk mengingatkan Anda tentang setiap upaya untuk mengakses rahasia selama jendela pemulihan. Untuk informasi selengkapnya, lihat Pantau kapan AWS Secrets Manager rahasia yang dijadwalkan untuk dihapus diakses.

Untuk menghapus rahasia, Anda harus memiliki `secretsmanager:ListSecrets` dan `secretsmanager:DeleteSecret` izin.

Manajer Rahasia menghasilkan entri CloudTrail log saat Anda menghapus rahasia. Untuk informasi selengkapnya, lihat the section called “Masuk dengan AWS CloudTrail ”.

Untuk menghapus rahasia (konsol)

1. Buka konsol Secrets Manager di https://console.aws.amazon.com/secretsmanager/.
2. Dalam daftar rahasia, pilih rahasia yang ingin Anda hapus.
3. Di bagian Detail rahasia, pilih Tind akan, lalu pilih H apus rahasia.
4. Di kotak dialog Non aktifkan rahasia dan jadwal penghapusan, di Masa tunggu, masukkan jumlah hari untuk menunggu sebelum penghapusan menjadi permanen. Manajer Rahasia melampirkan bidang yang dipanggil `DeletionDate` dan menetapkan bidang ke tanggal dan waktu saat ini, ditambah jumlah hari yang ditentukan untuk jendela pemulihan.
5. Pilih Jadwalkan penghapusan.

Untuk melihat rahasia yang dihapus

1. Buka konsol Secrets Manager di https://console.aws.amazon.com/secretsmanager/.
2. Pada halaman Rahasia, pilih Preferensi (⚙).
3. Di kotak dialog Preferensi, pilih Tampilkan rahasia yang dijadwalkan untuk dihapus, lalu pilih S impan.

Untuk menghapus rahasia replika

1. Buka konsol Secrets Manager di https://console.aws.amazon.com/secretsmanager/.
2. Pilih rahasia utama.
3. Di bagian Rahasia Replikasi, pilih rahasia replika.
4. Dari menu T indakan, pilih H apus Replika.

AWS CLI

Example Hapus rahasia

`delete-secret`Contoh berikut menghapus rahasia. Anda dapat memulihkan rahasia dengan `restore-secret` sampai tanggal dan waktu di bidang DeletionDate respons. Untuk menghapus

rahasia yang direplikasi ke wilayah lain, pertama-tama hapus replika dengan `remove-regions-from-replication`, lalu panggil. `delete-secret`

```
aws secretsmanager delete-secret \
    --secret-id MyTestSecret \
    --recovery-window-in-days 7
```

Example Hapus rahasia segera

`delete-secret`Contoh berikut menghapus rahasia segera tanpa jendela pemulihan. Anda tidak dapat memulihkan rahasia ini.

```
aws secretsmanager delete-secret \
    --secret-id MyTestSecret \
    --force-delete-without-recovery
```

Example Menghapus rahasia replika

`remove-regions-from-replication`Contoh berikut menghapus rahasia replika di eu-west-3. Untuk menghapus rahasia utama yang direplikasi ke wilayah lain, pertama-tama hapus replika dan kemudian panggil. `delete-secret`

```
aws secretsmanager remove-regions-from-replication \
    --secret-id MyTestSecret \
    --remove-replica-regions eu-west-3
```

AWS SDK

Untuk menghapus rahasia, gunakan `DeleteSecret` perintah. Untuk menghapus versi rahasia, gunakan `UpdateSecretVersionStage` perintah. Untuk menghapus replika, gunakan `StopReplicationToReplica` perintah. Lihat informasi yang lebih lengkap di the section called “AWS SDK”.

Kembalikan AWS Secrets Manager Rahasia

Manajer Rahasia menganggap rahasia yang dijadwalkan untuk dihapus tidak digunakan lagi dan Anda tidak dapat lagi mengaksesnya secara langsung. Setelah jendela pemulihan berlalu, Manajer Rahasia menghapus rahasia secara permanen. Setelah Manajer Rahasia menghapus rahasia, Anda

tidak dapat memulihkannya. Sebelum akhir jendela pemulihan, Anda dapat memulihkan rahasia dan membuatnya dapat diakses lagi. Ini menghapus `DeletionDate` bidang, yang membatalkan penghapusan permanen yang dijadwalkan.

Untuk memulihkan rahasia dan metadata di konsol, Anda harus memiliki `secretsmanager:ListSecrets` dan `secretsmanager:RestoreSecret` izin.

Manajer Rahasia menghasilkan entri CloudTrail log saat Anda memulihkan rahasia. Untuk informasi selengkapnya, lihat the section called “Masuk dengan AWS CloudTrail ”.

Untuk mengembalikan rahasia (konsol)

1. Buka konsol Secrets Manager di https://console.aws.amazon.com/secretsmanager/.
2. Dalam daftar rahasia, pilih rahasia yang ingin Anda pulihkan.

   Jika rahasia yang dihapus tidak muncul dalam daftar rahasia Anda, pilih Preferensi (⚙).
   Di kotak dialog Preferensi, pilih Tampilkan rahasia yang dijadwalkan untuk dihapus, lalu pilih S impan.
3. Pada halaman Detail rahasia, pilih Bat alkan penghapusan.
4. Di kotak dialog Batalkan penghapusan rahasia, pilih Bat alkan penghapusan.

AWS CLI

Example Kembalikan rahasia yang sebelumnya dihapus

`restore-secret`Contoh berikut mengembalikan rahasia yang sebelumnya dijadwalkan untuk dihapus.

```
aws secretsmanager restore-secret \
    --secret-id MyTestSecret
```

AWS SDK

Untuk mengembalikan rahasia yang ditandai untuk dihapus, gunakan `RestoreSecret` perintah. Lihat informasi yang lebih lengkap di the section called “AWS SDK”.

Menandai rahasia di AWS Secrets Manager

Di AWS Secrets Manager, Anda dapat menetapkan metadata ke rahasia Anda menggunakan tag. Tag adalah pasangan kunci-nilai yang Anda tentukan untuk rahasia. Tag membantu Anda mengelola AWS sumber daya dan mengatur data, termasuk informasi penagihan.

Dengan tag, Anda dapat:

- Mengelola, cari, dan filter rahasia dan sumber daya lainnya di AWS akun Anda
- Kontrol akses ke rahasia berdasarkan tag terlampir
- Melacak dan mengkategorikan pengeluaran yang terkait dengan rahasia atau proyek tertentu

Untuk informasi selengkapnya tentang menggunakan tag untuk mengontrol akses, lihatthe section called “Kontrol akses ke rahasia menggunakan tag”.

Untuk mempelajari tentang tag alokasi biaya, lihat Menggunakan tag alokasi AWS biaya di Panduan AWS Billing Pengguna.

Untuk informasi tentang kuota tag dan batasan penamaan, lihat Kuota layanan untuk Penandaan di Panduan Refer AWS ensi Umum. Tag peka huruf besar/kecil.

Manajer Rahasia menghasilkan entri CloudTrail log saat Anda menandai atau menghapus tag rahasia. Untuk informasi selengkapnya, lihat the section called “Masuk dengan AWS CloudTrail ”.

> **Tip**
>
> Gunakan skema penandaan yang konsisten di semua AWS sumber daya Anda. Untuk praktik terbaik, lihat whitepaper Penandaan Praktik Terbaik.

Tinjau dasar-dasar tag

Anda dapat menemukan rahasia dengan tag di konsol, AWS CLI, dan SDK. AWS juga menyediakan alat Grup Sumber Daya untuk membuat konsol kustom yang mengkonsolidasikan dan mengatur sumber daya Anda berdasarkan tag mereka. Untuk menemukan rahasia dengan tag tertentu, lihatthe section called “Temukan rahasia”.

Anda dapat menggunakan konsol Secrets Manager AWS CLI, atau API Manajer Rahasia untuk:

- Buat rahasia dengan tag
- Menambahkan tag ke rahasia
- Buat daftar tag untuk rahasia Anda
- Hapus tag dari rahasia

Anda dapat menggunakan tag untuk mengkategorikan rahasia Anda. Misalnya, Anda dapat mengkategorikan rahasia berdasarkan tujuan, pemilik, atau lingkungan. Karena Anda menentukan kunci dan nilai untuk setiap tanda, Anda dapat membuat serangkaian kategori khusus untuk memenuhi kebutuhan spesifik Anda. Berikut adalah beberapa contoh tanda:

- `Project: Project name`
- `Owner: Name`
- `Purpose: Load testing`
- `Application: Application name`
- `Environment: Production`

Lacak biaya menggunakan penandaan

Anda dapat menggunakan tag untuk mengkategorikan dan melacak AWS biaya Anda. Saat Anda menerapkan tag ke AWS sumber daya, termasuk rahasia, laporan alokasi AWS biaya Anda mencakup penggunaan dan biaya yang digabungkan berdasarkan tag. Anda dapat menerapkan tag yang mewakili kategori bisnis (seperti pusat biaya, nama aplikasi, atau pemilik) untuk mengatur biaya Anda di berbagai layanan. Untuk informasi selengkapnya, lihat Menggunakan Tanda Alokasi Biaya untuk Laporan Penagihan Khusus dalam Panduan Pengguna AWS Billing .

Memahami batasan tag

Batasan berikut berlaku untuk tanda.

Batasan dasar

- Jumlah maksimum tag per sumber daya (rahasia) adalah 50.
- Kunci dan nilai tag peka terhadap huruf besar dan kecil.
- Anda tidak dapat mengubah atau mengedit tag untuk rahasia yang dihapus.

Batasan kunci tanda

- Setiap kunci tanda harus unik. Jika Anda menambahkan tanda dengan kunci yang sudah digunakan, tanda baru akan menimpa pasangan nilai-kunci yang sudah ada.
- Anda tidak dapat memulai kunci tag dengan `aws:` karena awalan ini disediakan untuk digunakan oleh AWS. AWS membuat tag yang dimulai dengan awalan ini atas nama Anda, tetapi Anda tidak dapat mengedit atau menghapusnya.
- Kunci tanda harus memiliki panjang antara 1 dan 128 karakter Unicode.
- Kunci tanda harus terdiri dari karakter berikut: huruf Unicode, digit, spasi, dan karakter khusus berikut: `_ . / = + - @`.

Batasan nilai tanda

- Nilai tag harus antara 0 dan 256 karakter Unicode panjangnya.
- Nilai tanda dapat kosong. Jika tidak, nilai tanda harus terdiri dari karakter berikut: huruf Unicode, digit, spasi, dan salah satu karakter khusus berikut: `_ . / = + - @`.

Tandai rahasia menggunakan konsol Secrets Manager

Anda dapat mengelola tag untuk rahasia Anda menggunakan konsol Manajer Rahasia.

Untuk mengakses fitur penandaan, lakukan hal berikut:

1. Buka konsol Secrets Manager.
2. Di bilah navigasi, pilih Wilayah pilihan Anda.
3. Pada halaman Rahasia, pilih rahasia.

Untuk melihat tag untuk rahasia

- Pada halaman Detail Rahasia, pilih tab Tag.

Untuk membuat rahasia dengan tag

- Ikuti langkah-langkah di Buat rahasia.

Untuk menambahkan atau mengedit tag untuk rahasia

1. Pada halaman Detail Rahasia, pilih tab Tag dan kemudian pilih Edit tag.
2. Masukkan kunci tag di bidang Kunci. Secara opsional, masukkan nilai tag di bidang Nilai.
3. Pilih Simpan. Tag baru atau diperbarui muncul dalam daftar tag.

> **Note**
>
> Jika tombol Simpan tidak diaktifkan, kunci tag atau nilai mungkin tidak memenuhi batasan tag. Untuk informasi selengkapnya, lihat Memahami batasan tag.

Untuk menghapus tag dari rahasia

1. Pada halaman Detail Rahasia, pilih tab Tag, lalu pilih ikon Hapus di sebelah tag yang ingin Anda hapus.
2. Pilih Simpan untuk mengonfirmasi penghapusan, atau pilih Batalkan untuk membatalkan.

Tandai rahasia menggunakan AWS CLI

AWS CLI contoh

Example Menambahkan tag ke rahasia

`tag-resource`Contoh berikut menunjukkan cara melampirkan tag dengan sintaks steno.

```
aws secretsmanager tag-resource \
            --secret-id MyTestSecret \
            --tags Key=FirstTag,Value=FirstValue
```

Example Menambahkan beberapa tag ke rahasia

`tag-resource`Contoh berikut melampirkan dua tag nilai kunci ke rahasia.

```
aws secretsmanager tag-resource \
            --secret-id MyTestSecret \
            --tags '[{"Key": "FirstTag", "Value": "FirstValue"}, {"Key": "SecondTag",
 "Value": "SecondValue"}]'
```

Example Hapus tag dari rahasia

`untag-resource`Contoh berikut menghapus dua tag dari rahasia. Untuk setiap tag, kunci dan nilai dihapus.

```
aws secretsmanager untag-resource \
            --secret-id MyTestSecret \
            --tag-keys '[ "FirstTag", "SecondTag"]'
```

Tandai rahasia menggunakan API Manajer Rahasia

Anda dapat menambahkan, membuat daftar, dan menghapus tag menggunakan API Manajer Rahasia. Untuk contoh, lihat dokumentasi berikut:

- ListSecrets: Gunakan `ListSecrets` untuk melihat tag yang diterapkan pada rahasia
- TagResource: Tambahkan tag ke rahasia
- Untag: Hapus tag dari rahasia

Tandai rahasia menggunakan Manajer Rahasia AWS SDK

Untuk mengubah tag rahasia Anda, gunakan operasi API berikut:

- ListSecrets: Gunakan `ListSecrets` untuk melihat tag yang diterapkan pada rahasia
- `TagResource`: Tambahkan tag ke rahasia
- `UntagResource`: Hapus tag dari rahasia

Untuk informasi selengkapnya tentang menggunakan SDK, lihatthe section called “AWS SDK”.

Mereplikasi AWS Secrets Manager Rahasia di seluruh Wilayah

Anda dapat mereplikasi rahasia Anda dalam beberapa Wilayah AWS untuk mendukung aplikasi yang tersebar di seluruh Wilayah tersebut untuk memenuhi akses Regional dan persyaratan latensi rendah. Jika nantinya perlu, Anda dapat mempromosikan rahasia replika ke mandiri dan kemudian mengaturnya untuk replikasi secara independen. Secrets Manager mereplikasi data rahasia dan metadata terenkripsi seperti tag dan kebijakan sumber daya di seluruh Wilayah yang ditentukan.

ARN untuk rahasia yang direplikasi sama dengan rahasia utama kecuali untuk Wilayah, misalnya:

- Rahasia utama: `arn:aws:secretsmanager:Region1:123456789012:secret:MySecret-a1b2c3`
- Rahasia replika: `arn:aws:secretsmanager:Region2:123456789012:secret:MySecret-a1b2c3`

Untuk informasi harga untuk rahasia replika, lihat AWS Secrets Manager Harga.

Saat Anda menyimpan kredentif database untuk database sumber yang direplikasi ke Wilayah lain, rahasia berisi informasi koneksi untuk database sumber. Jika Anda kemudian mereplikasi rahasia, replika adalah salinan rahasia sumber dan berisi informasi koneksi yang sama. Anda dapat menambahkan key/value pasangan tambahan ke rahasia untuk informasi koneksi regional.

Jika Anda mengaktifkan rotasi untuk rahasia utama Anda, Manajer Rahasia akan memutar rahasia di Wilayah utama, dan nilai rahasia baru menyebar ke semua rahasia replika terkait. Anda tidak perlu mengelola rotasi secara individual untuk semua rahasia replika.

Anda dapat mereplikasi rahasia di semua Wil AWS ayah yang diaktifkan. Namun, jika Anda menggunakan Manajer Rahasia di Wil AWS ayah khusus seperti AWS GovCloud (US) atau Wilayah China, Anda hanya dapat mengonfigurasi rahasia dan replika dalam Wil AWS ayah khusus ini. Anda tidak dapat mereplikasi rahasia di Wil AWS ayah yang diaktifkan ke Wilayah khusus atau mereplikasi rahasia dari wilayah khusus ke wilayah komersial.

Sebelum Anda dapat mereplikasi rahasia ke Wilayah lain, Anda harus mengaktifkan Wilayah itu. Untuk informasi selengkapnya, lihat M engelola AWS Wilayah.

Dimungkinkan untuk menggunakan rahasia di beberapa Wilayah tanpa mereplikasinya dengan memanggil titik akhir Manajer Rahasia di Wilayah tempat rahasia disimpan. Untuk daftar titik

akhir, lihatthe section called “Titik akhir Manajer Rahasia”. Untuk menggunakan replikasi untuk meningkatkan ketahanan beban kerja Anda, lihat Ar sitektur Pemulihan Bencana (DR) pada AWS, Bagian I: Strategi untuk Pemulihan di Cloud.

Manajer Rahasia menghasilkan entri CloudTrail log saat Anda mereplikasi rahasia. Untuk informasi selengkapnya, lihat the section called “Masuk dengan AWS CloudTrail ”.

Untuk mereplikasi rahasia ke Wilayah lain (konsol)

1. Buka konsol Secrets Manager di https://console.aws.amazon.com/secretsmanager/.
2. Dari daftar rahasia, pilih rahasia Anda.
3. Pada halaman detail rahasia, pada tab Replikasi, lakukan salah satu hal berikut:
   - Jika rahasia Anda tidak direplikasi, pilih Rep likasi rahasia.
   - Jika rahasia Anda direplikasi, di bagian Rahasia Rep likasi, pilih Tam bah Wilayah.
4. Di kotak dialog Tambahkan wilayah replika, lakukan hal berikut:
   a. Untuk AWS Wilayah, pilih Wilayah yang ingin Anda tiru rahasianya.
   b. (Opsional) Untuk kunci enkripsi, pilih kunci KMS untuk mengenkripsi rahasia. Kuncinya harus di Wilayah replika.
   c. (Opsional) Untuk menambahkan Wilayah lain, pilih Tambah wilayah lainnya.
   d. Pilih Rep likasi.

   Anda kembali ke halaman detail rahasia. Di bagian rahasia Replikasi, status Replikasi ditampilkan untuk setiap Wilayah.

AWS CLI

Example Mereplikasi rahasia ke wilayah lain

`replicate-secret-to-regions`Contoh berikut mereplikasi rahasia ke eu-west-3. Replika dienkripsi dengan kunci ter AWS kelolaaws/secretsmanager.

```
aws secretsmanager replicate-secret-to-regions \
        --secret-id MyTestSecret \
        --add-replica-regions Region=eu-west-3
```

Example Buat rahasia dan tiru

Contoh berikut membuat rahasia dan mereplikasinya ke eu-west-3. Replika dienkripsi dengan Kunci yang dikelola AWS aws/secretsmanager.

```
aws secretsmanager create-secret \
    --name MyTestSecret \
    --description "My test secret created with the CLI." \
    --secret-string "{\"user\":\"diegor\",\"password\":\"EXAMPLE-PASSWORD\"}"
    --add-replica-regions Region=eu-west-3
```

AWS SDK

Untuk mereplikasi rahasia, gunakan `ReplicateSecretToRegions` perintah. Lihat informasi yang lebih lengkap di the section called “AWS SDK”.

Promosikan rahasia replika ke rahasia mandiri di AWS Secrets Manager

Rahasia replika adalah rahasia yang direplikasi dari yang utama di yang lain Wilayah AWS. Ini memiliki nilai rahasia dan metadata yang sama dengan yang utama, tetapi dapat dienkripsi dengan kunci KMS yang berbeda. Rahasia replika tidak dapat diperbarui secara independen dari rahasia utamanya, kecuali untuk kunci enkripsinya. Mempromosikan rahasia replika memutuskan rahasia replika dari rahasia utama dan membuat rahasia replika menjadi rahasia mandiri. Perubahan pada rahasia utama tidak akan mereplikasi ke rahasia mandiri.

Anda mungkin ingin mempromosikan rahasia replika ke rahasia mandiri sebagai solusi pemulihan bencana jika rahasia utama menjadi tidak tersedia. Atau Anda mungkin ingin mempromosikan replika ke rahasia mandiri jika Anda ingin mengaktifkan rotasi untuk replika.

Jika Anda mempromosikan replika, pastikan untuk memperbarui aplikasi yang sesuai untuk menggunakan rahasia mandiri.

Manajer Rahasia menghasilkan entri CloudTrail log saat Anda mempromosikan rahasia. Untuk informasi selengkapnya, lihat the section called “Masuk dengan AWS CloudTrail ”.

Untuk mempromosikan rahasia replika (konsol)

1. Masuk ke Manajer Rahasia di https://console.aws.amazon.com/secretsmanager/.

2. Arahkan ke wilayah replika.
3. Pada halaman Rahasia, pilih rahasia replika.
4. Pada halaman detail rahasia replika, pilih Promosikan ke rahasia mandiri.
5. Di kotak dialog Promosikan replika ke rahasia mandiri, masukkan Wilayah dan kemudian pilih Promosikan replika.

AWS CLI

Example Promosikan rahasia replika ke primer

`stop-replication-to-replica`Contoh berikut menghapus tautan antara rahasia replika ke primer. Rahasia replika dipromosikan menjadi rahasia utama di wilayah replika. Anda harus menelepon `stop-replication-to-replica` dari dalam wilayah replika.

```
aws secretsmanager stop-replication-to-replica \
    --secret-id MyTestSecret
```

AWS SDK

Untuk mempromosikan replika ke rahasia mandiri, gunakan `StopReplicationToReplica` perintah. Anda harus memanggil perintah ini dari Wilayah rahasia replika. Lihat informasi yang lebih lengkap di the section called “AWS SDK”.

Mencegah AWS Secrets Manager replikasi

Karena rahasia dapat direplikasi menggunakan `ReplicateSecretToRegions`atau ketika mereka dibuat menggunakan `CreateSecret`, jika Anda ingin mencegah pengguna mereplikasi rahasia, kami sarankan Anda mencegah tindakan yang berisi parameter. `AddReplicaRegions` Anda dapat menggunakan `Condition` pernyataan dalam kebijakan izin untuk hanya mengizinkan tindakan yang tidak menambahkan wilayah replika. Lihat contoh kebijakan berikut untuk pernyataan Kondisi yang dapat Anda gunakan.

Example Mencegah izin replikasi

Contoh kebijakan berikut menunjukkan cara mengizinkan semua tindakan yang tidak menambahkan wilayah replika. Ini mencegah pengguna mereplikasi rahasia melalui keduanya `ReplicateSecretToRegions` dan`CreateSecret`.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "secretsmanager:*",
      "Resource": "*",
      "Condition": {
        "Null": {
          "secretsmanager:AddReplicaRegions": "true"
        }
      }
    }
  ]
}
```

Example Izinkan izin replikasi hanya untuk Wilayah tertentu

Kebijakan berikut menunjukkan cara mengizinkan semua hal berikut:

- Buat rahasia tanpa replikasi
- Buat rahasia dengan replikasi ke Wilayah hanya di Amerika Serikat dan Kanada
- Replikasi rahasia ke Wilayah hanya di Amerika Serikat dan Kanada

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:CreateSecret",
        "secretsmanager:ReplicateSecretToRegions"
      ],
      "Resource": "*",
      "Condition": {
```

```
        "ForAllValues:StringLike": {
          "secretsmanager:AddReplicaRegions": [
            "us-*",
            "ca-*"
          ]
        }
      }
    }
  ]
}
```

Pemecahan Masalah AWS Secrets Manager replikasi

AWS Secrets Manager replikasi mungkin gagal karena berbagai alasan. Untuk memeriksa mengapa rahasia gagal direplikasi, Anda dapat melakukan salah satu hal berikut:

- Panggil operasi `DescribeSecret` API
- Tinjau AWS CloudTrail acara

Ketika replikasi gagal:

- Jika tidak ada versi rahasia yang dapat digunakan, Manajer Rahasia menghapus rahasia dari Wilayah replika.
- Jika ada versi rahasia yang berhasil direplikasi, versi tersebut tetap berada di Wilayah replika sampai Anda menghapusnya secara eksplisit menggunakan operasi `RemoveRegionsFromReplication` API.

Bagian berikut menjelaskan beberapa alasan umum kegagalan replikasi.

Rahasia dengan nama yang sama ada di Wilayah yang dipilih

Untuk mengatasi masalah ini, Anda dapat mengganti rahasia nama duplikat di Wilayah replika. Coba ulang replikasi, lalu di kotak dialog C oba replikasi ulang, pilih Overwrite.

Tidak ada izin yang tersedia pada kunci KMS untuk menyelesaikan replikasi

Manajer Rahasia pertama-tama mendekripsi rahasia sebelum mengenkripsi ulang dengan kunci KMS baru di Wilayah replika. Jika Anda tidak memiliki `kms:Decrypt` izin untuk kunci enkripsi di Wilayah

utama, Anda akan mengalami kesalahan ini. Untuk mengenkripsi rahasia yang direplikasi dengan kunci KMS selain`aws/secretsmanager`, Anda perlu `kms:GenerateDataKey` dan `kms:Encrypt` ke kunci. Lihat the section called “Izin untuk kunci KMS”.

Kunci KMS dinonaktifkan atau tidak ditemukan

Jika kunci enkripsi di Wilayah utama dinonaktifkan atau dihapus, Manajer Rahasia tidak dapat mereplikasi rahasia tersebut. Kesalahan ini dapat terjadi bahkan jika Anda telah mengubah kunci enkripsi, jika rahasia memiliki versi berlabel khusus yang dienkripsi dengan kunci enkripsi yang dinonaktifkan atau dihapus. Untuk informasi tentang bagaimana Secrets Manager melakukan enkripsi, lihatthe section called “Enkripsi rahasia dan dekripsi”. Untuk mengatasi masalah ini, Anda dapat membuat ulang versi rahasia sehingga Manajer Rahasia mengenkripsinya dengan kunci enkripsi saat ini. Untuk informasi selengkapnya, lihat Meng ubah kunci enkripsi untuk rahasia. Kemudian coba replikasi lagi.

```
aws secretsmanager put-secret-value \
  --secret-id testDescriptionUpdate \
  --secret-string "SecretValue" \
  --version-stages "MyCustomLabel"
```

Anda belum mengaktifkan Wilayah tempat replikasi terjadi

Untuk informasi tentang cara mengaktifkan Wilayah, lihat M engelola Wil AWS ayah. dalam Panduan Referensi Manajemen AWS Akun.

Dapatkan rahasia dari AWS Secrets Manager

Manajer Rahasia menghasilkan entri CloudTrail log saat Anda mengambil rahasia. Untuk informasi selengkapnya, lihat the section called “Masuk dengan AWS CloudTrail ”.

Anda dapat mengambil nilai rahasia menggunakan:

- Dapatkan nilai rahasia Secrets Manager menggunakan Java
- Dapatkan nilai rahasia Secrets Manager menggunakan Python
- Dapatkan nilai rahasia Secrets Manager menggunakan .NET
- Dapatkan nilai rahasia Secrets Manager menggunakan Go
- Dapatkan nilai rahasia Secrets Manager menggunakan Rust
- Gunakan AWS Secrets Manager rahasia di Amazon Elastic Kubernetes Service
- Gunakan AWS Secrets Manager rahasia dalam AWS Lambda fungsi
- Menggunakan AWS Penyedia Kredensia Beban Kerja
- Gunakan rahasia dengan aman dengan Agen Pengkodean AI
- Dapatkan nilai rahasia Manajer Rahasia menggunakan C ++ AWS SDK
- Dapatkan nilai rahasia Manajer Rahasia menggunakan JavaScript AWS SDK
- Dapatkan nilai rahasia Manajer Rahasia menggunakan Kotlin AWS SDK
- Dapatkan nilai rahasia Manajer Rahasia menggunakan PHP AWS SDK
- Dapatkan nilai rahasia Manajer Rahasia menggunakan Ruby AWS SDK
- Dapatkan nilai rahasia menggunakan AWS CLI
- Dapatkan nilai rahasia menggunakan AWS konsol
- Gunakan AWS Secrets Manager rahasia di AWS Batch
- Dapatkan AWS Secrets Manager rahasia di sumber CloudFormation daya
- Gunakan AWS Secrets Manager rahasia dalam GitHub pekerjaan
- Gunakan AWS Secrets Manager di GitLab
- Gunakan AWS Secrets Manager rahasia di AWS IoT Greengrass
- Gunakan AWS Secrets Manager rahasia di Parameter Store

Dapatkan nilai rahasia Secrets Manager menggunakan Java

Dalam aplikasi, Anda dapat mengambil rahasia Anda dengan menelepon `GetSecretValue` atau `BatchGetSecretValue` di salah satu AWS SDK. Namun, kami menyarankan Anda menyimpan nilai rahasia Anda dengan menggunakan caching sisi klien. Rahasia caching meningkatkan kecepatan dan mengurangi biaya Anda.

Untuk terhubung ke database menggunakan kredensyal secara rahasia, Anda dapat menggunakan driver Secrets Manager SQL Connection, yang membungkus driver JDBC dasar. Ini juga menggunakan caching sisi klien, sehingga dapat mengurangi biaya untuk memanggil Secrets Manager API.

Topik

- Dapatkan nilai rahasia Secrets Manager menggunakan Java dengan caching sisi klien
- Connect ke database SQL menggunakan JDBC dengan kredensi dalam rahasia AWS Secrets Manager
- Dapatkan nilai rahasia Secrets Manager menggunakan Java AWS SDK

Dapatkan nilai rahasia Secrets Manager menggunakan Java dengan caching sisi klien

Saat Anda mengambil rahasia, Anda dapat menggunakan komponen Java-based caching Secrets Manager untuk men-cache untuk digunakan di masa mendatang. Mengambil rahasia yang di-cache lebih cepat daripada mengambilnya dari Secrets Manager. Karena ada biaya untuk memanggil Secrets Manager API, menggunakan cache dapat mengurangi biaya Anda. Untuk semua cara Anda dapat mengambil rahasia, lihatDapatkan rahasia.

Kebijakan cache adalah Least Recently Used (LRU), jadi ketika cache harus membuang rahasia, ia membuang rahasia yang paling jarang digunakan. Secara default, cache menyegarkan rahasia setiap jam. Anda dapat mengonfigurasi seberapa sering rahasia disegarkan dalam cache, dan Anda dapat menghubungkan ke pengambilan rahasia untuk menambahkan lebih banyak fungsionalitas.

Cache tidak memaksa pengumpulan sampah setelah referensi cache dibebaskan. Implementasi cache tidak termasuk pembatalan cache. Implementasi cache difokuskan di sekitar cache itu sendiri, dan tidak dikeraskan atau difokuskan keamanan. Jika Anda memerlukan keamanan tambahan seperti mengenkripsi item dalam cache, gunakan antarmuka dan metode abstrak yang disediakan.

Untuk menggunakan komponen, Anda harus memiliki yang berikut:

- Java 8 atau lingkungan pengembangan yang lebih tinggi. Lihat Unduhan Java SE di situs web Oracle.

Untuk mengunduh kode sumber, lihat Secrets Manager Java-based caching komponen klien di GitHub.

Untuk menambahkan komponen ke proyek Anda, dalam file pom.xml Maven Anda, sertakan dependensi berikut. Untuk informasi lebih lanjut tentang Maven, lihat Panduan Memulai di situs web Apache Maven Project.

```
<dependency>
  <groupId>com.amazonaws.secretsmanager</groupId>
  <artifactId>aws-secretsmanager-caching-java</artifactId>
  <version>1.0.2</version>
</dependency>
```

Izin yang diperlukan:

- `secretsmanager:DescribeSecret`
- `secretsmanager:GetSecretValue`

Untuk informasi selengkapnya, lihat Referensi izin.

Referensi

- SecretCache
- SecretCacheConfiguration
- SecretCacheHook

Example Ambil rahasia

Contoh kode berikut menunjukkan fungsi Lambda yang mengambil string rahasia. Ini mengikuti praktik terbaik untuk membuat instance cache di luar penangan fungsi, sehingga tidak terus memanggil API jika Anda memanggil fungsi Lambda lagi.

```
package com.amazonaws.secretsmanager.caching.examples;
```

```
    import com.amazonaws.services.lambda.runtime.Context;
    import com.amazonaws.services.lambda.runtime.RequestHandler;
    import com.amazonaws.services.lambda.runtime.LambdaLogger;

    import com.amazonaws.secretsmanager.caching.SecretCache;

    public class SampleClass implements RequestHandler<String, String> {

         private final SecretCache cache  = new SecretCache();

         @Override public String handleRequest(String secretId,  Context context) {
             final String secret  = cache.getSecretString(secretId);

            // Use the secret, return success;

        }
    }
```

SecretCache

Cache dalam memori untuk rahasia yang diminta dari Secrets Manager. Anda menggunakan the section called “mendapatkan SecretString” atau the section called “mendapatkan SecretBinary” untuk mengambil rahasia dari cache. Anda dapat mengkonfigurasi pengaturan cache dengan meneruskan the section called “SecretCacheConfiguration” objek di konstruktor.

Untuk informasi selengkapnya, termasuk contoh, lihatthe section called “Java dengan caching sisi klien”.

Konstruktor

`public SecretCache()`

Konstruktor default untuk `SecretCache` objek.

`public SecretCache(AWSSecretsManagerClientBuilder builder)`

Membangun cache baru menggunakan klien Secrets Manager yang dibuat menggunakan yang disediakan `AWSSecretsManagerClientBuilder`. Gunakan konstruktor ini untuk menyesuaikan klien Secrets Manager, misalnya untuk menggunakan Region atau endpoint tertentu.

`public SecretCache(AWSSecretsManager client)`

Membangun cache rahasia baru menggunakan yang disediakan `AWSSecretsManagerClient`. Gunakan konstruktor ini untuk menyesuaikan klien Secrets Manager, misalnya untuk menggunakan Region atau endpoint tertentu.

`public SecretCache(SecretCacheConfiguration config)`

Membangun cache rahasia baru menggunakan yang disediakanthe section called "SecretCacheConfiguration".

Metode

mendapatkan SecretString

`public String getSecretString(final String secretId)`

Mengambil rahasia string dari Secrets Manager. Mengembalikan `String`.

mendapatkan SecretBinary

`public ByteBuffer getSecretBinary(final String secretId)`

Mengambil rahasia biner dari Secrets Manager. Mengembalikan `ByteBuffer`.

RefreshNow

`public boolean refreshNow(final String secretId) throws InterruptedException`

Memaksa cache untuk menyegarkan. Kembali `true` jika refresh selesai tanpa kesalahan, jika tidak`false`.

tutup

`public void close()`

Menutup cache.

SecretCacheConfiguration

Opsi konfigurasi cache untukthe section called "SecretCache", seperti ukuran cache maks dan Time to Live (TTL) untuk rahasia cache.

Konstruktor

`public SecretCacheConfiguration`

Konstruktor default untuk `SecretCacheConfiguration` objek.

Metode

getClient

`public AWSSecretsManager getClient()`

Mengembalikan `AWSSecretsManagerClient`bahwa cache mengambil rahasia dari.

SetClient

`public void setClient(AWSSecretsManager client)`

Menetapkan `AWSSecretsManagerClient`klien tempat cache mengambil rahasia dari.

mendapatkan CacheHook

`public SecretCacheHook getCacheHook()`

Mengembalikan the section called “SecretCacheHook” antarmuka yang digunakan untuk mengaitkan pembaruan cache.

set CacheHook

`public void setCacheHook(SecretCacheHook cacheHook)`

Mengatur the section called “SecretCacheHook” antarmuka yang digunakan untuk mengaitkan pembaruan cache.

mendapatkan MaxCacheSize

`public int getMaxCacheSize()`

Mengembalikan ukuran cache maksimum. Defaultnya adalah 1024 rahasia.

set MaxCacheSize

`public void setMaxCacheSize(int maxCacheSize)`

Menetapkan ukuran cache maksimum. Defaultnya adalah 1024 rahasia.

dapatkan CacheItem TTL

```
public long getCacheItemTTL()
```

Mengembalikan TTL dalam milidetik untuk item cache. Ketika rahasia yang di-cache melebihi TTL ini, cache mengambil salinan rahasia baru dari file. `AWSSecretsManagerClient` Defaultnya adalah 1 jam dalam milidetik.

Cache menyegarkan rahasia secara serempak ketika rahasia diminta setelah TTL. Jika penyegaran sinkron gagal, cache mengembalikan rahasia basi.

set CacheItem TTL

```
public void setCacheItemTTL(long cacheItemTTL)
```

Mengatur TTL dalam milidetik untuk item yang di-cache. Ketika rahasia yang di-cache melebihi TTL ini, cache mengambil salinan rahasia baru dari file. `AWSSecretsManagerClient` Defaultnya adalah 1 jam dalam milidetik.

mendapatkan VersionStage

```
public String getVersionStage()
```

Mengembalikan versi rahasia yang ingin Anda cache. Untuk informasi selengkapnya, lihat Versi rahasia. Defaultnya adalah `"AWSCURRENT"`.

set VersionStage

```
public void setVersionStage(String versionStage)
```

Menetapkan versi rahasia yang ingin Anda cache. Untuk informasi selengkapnya, lihat Versi rahasia. Nilai default-nya `"AWSCURRENT"`.

SecretCacheConfiguration denganKlien

```
public SecretCacheConfiguration withClient(AWSSecretsManager client)
```

Menetapkan `AWSSecretsManagerClient`untuk mengambil rahasia dari. Mengembalikan `SecretCacheConfiguration` objek diperbarui dengan pengaturan baru.

SecretCacheConfiguration dengan CacheHook

```
public SecretCacheConfiguration withCacheHook(SecretCacheHook cacheHook)
```

Mengatur antarmuka yang digunakan untuk mengaitkan cache dalam memori. Mengembalikan `SecretCacheConfiguration` objek diperbarui dengan pengaturan baru.

SecretCacheConfiguration dengan MaxCacheSize

```
public SecretCacheConfiguration withMaxCacheSize(int maxCacheSize)
```

Menetapkan ukuran cache maksimum. Mengembalikan `SecretCacheConfiguration` objek diperbarui dengan pengaturan baru.

SecretCacheConfiguration dengan CacheItem TTL

```
public SecretCacheConfiguration withCacheItemTTL(long cacheItemTTL)
```

Mengatur TTL dalam milidetik untuk item yang di-cache. Ketika rahasia yang di-cache melebihi TTL ini, cache mengambil salinan rahasia baru dari file. `AWSSecretsManagerClient` Defaultnya adalah 1 jam dalam milidetik. Mengembalikan `SecretCacheConfiguration` objek diperbarui dengan pengaturan baru.

SecretCacheConfiguration dengan VersionStage

```
public SecretCacheConfiguration withVersionStage(String versionStage)
```

Menetapkan versi rahasia yang ingin Anda cache. Untuk informasi selengkapnya, lihat Versi rahasia. Mengembalikan `SecretCacheConfiguration` objek diperbarui dengan pengaturan baru.

SecretCacheHook

Antarmuka untuk menghubungkan the section called “SecretCache” ke dalam untuk melakukan tindakan pada rahasia yang disimpan dalam cache.

menempatkan

```
Object put(final Object o)
```

Siapkan objek untuk disimpan dalam cache.

Mengembalikan objek untuk menyimpan dalam cache.

memperoleh

```
Object get(final Object cachedObject)
```

Turunkan objek dari objek yang di-cache.

Mengembalikan objek untuk kembali dari cache

Connect ke database SQL menggunakan JDBC dengan kredensi dalam rahasia AWS Secrets Manager

Dalam aplikasi Java, Anda dapat menggunakan driver Secrets Manager SQL Connection untuk terhubung ke MySQL, PostgreSQL, Oracle, MSSQLServer, Db2, dan database Redshift menggunakan kredensil yang disimpan di Secrets Manager. Setiap driver membungkus driver JDBC dasar, sehingga Anda dapat menggunakan panggilan JDBC untuk mengakses database Anda. Namun, alih-alih memberikan nama pengguna dan kata sandi untuk koneksi, Anda memberikan ID rahasia. Pengemudi memanggil Secrets Manager untuk mengambil nilai rahasia, dan kemudian menggunakan kredensil dalam rahasia untuk terhubung ke database. Driver juga menyimpan kredensialnya menggunakan pustaka caching sisi klien Java, sehingga koneksi future tidak memerlukan panggilan ke Secrets Manager. Secara default, cache diperbarui setiap jam dan juga ketika rahasia diputar. Untuk mengkonfigurasi cache, lihatthe section called “SecretCacheConfiguration”.

Anda dapat mengunduh kode sumber dari GitHub.

Untuk menggunakan driver Secrets Manager SQL Connection:

- Aplikasi Anda harus di Java 8 atau lebih tinggi.
- Rahasia Anda harus salah satu dari yang berikut:
  - Sebuah rahasia database dalam struktur JSON diharapkan. Untuk memeriksa format, di konsol Secrets Manager, lihat rahasia Anda dan pilih Ambil nilai rahasia. Atau, di AWS CLI, panggil get-secret-value.
  - Rahasia yang dikelola Amazon RDS. Untuk jenis rahasia ini, Anda harus menentukan titik akhir dan port saat Anda membuat koneksi.
  - Rahasia yang dikelola Amazon Redshift. Untuk jenis rahasia ini, Anda harus menentukan titik akhir dan port saat Anda membuat koneksi.

Jika database Anda direplikasi ke Wilayah lain, untuk menyambung ke database replika di Wilayah lain, Anda menentukan titik akhir dan port regional saat Anda membuat koneksi. Anda dapat menyimpan informasi koneksi regional secara rahasia sebagai key/value pasangan tambahan, dalam parameter Penyimpanan Parameter SSM, atau dalam konfigurasi kode Anda.

Untuk menambahkan driver ke proyek Anda, dalam file build Maven Andapom.xml, tambahkan dependensi berikut untuk driver. Untuk informasi selengkapnya, lihat Secrets Manager SQL Connection Library di situs web Maven Central Repository.

```
<dependency>
    <groupId>com.amazonaws.secretsmanager</groupId>
    <artifactId>aws-secretsmanager-jdbc</artifactId>
    <version>1.0.12</version>
</dependency>
```

Pengemudi menggunakan rantai penyedia kredensi default. Jika Anda menjalankan driver di Amazon EKS, itu mungkin mengambil kredensil node yang dijalankannya alih-alih peran akun layanan. Untuk mengatasinya, tambahkan versi 1 com.amazonaws:aws-java-sdk-sts ke file proyek Gradle atau Maven Anda sebagai dependensi.

Untuk mengatur URL endpoint AWS PrivateLink DNS dan wilayah dalam file: secretsmanager.properties

```
drivers.vpcEndpointUrl = endpoint URL
drivers.vpcEndpointRegion = endpoint region
```

Untuk mengganti wilayah primer, atur variabel AWS_SECRET_JDBC_REGION lingkungan atau buat perubahan berikut ke secretsmanager.properties file:

```
drivers.region = region
```

Izin yang diperlukan:

- secretsmanager:DescribeSecret
- secretsmanager:GetSecretValue

Untuk informasi selengkapnya, lihat Referensi izin.

Contoh:

- Membuat koneksi ke database
- Buat koneksi dengan menentukan titik akhir dan port
- Gunakan penyatuan koneksi c3p0 untuk membuat koneksi
- Gunakan penyatuan koneksi c3p0 untuk membuat koneksi dengan menentukan titik akhir dan port

Membuat koneksi ke database

Contoh berikut menunjukkan cara membuat koneksi ke database menggunakan kredensil dan informasi koneksi secara rahasia. Setelah Anda memiliki koneksi, Anda dapat menggunakan panggilan JDBC untuk mengakses database. Untuk informasi selengkapnya, lihat JDBC Basics di situs web dokumentasi Java.

MySQL

```
// Load the JDBC driver
Class.forName( "com.amazonaws.secretsmanager.sql.AWSSecretsManagerMySQLDriver" ).newInstance

// Retrieve the connection info from the secret using the secret ARN
String URL = "secretId";

// Populate the user property with the secret ARN to retrieve user and password from
 the secret
Properties info = new Properties( );
info.put( "user", "secretId" );

// Establish the connection
conn = DriverManager.getConnection(URL, info);
```

PostgreSQL

```
// Load the JDBC driver
Class.forName( "com.amazonaws.secretsmanager.sql.AWSSecretsManagerPostgreSQLDriver" ).newIns

// Retrieve the connection info from the secret using the secret ARN
String URL = "secretId";

// Populate the user property with the secret ARN to retrieve user and password from
 the secret
Properties info = new Properties( );
info.put( "user", "secretId" );

// Establish the connection
conn = DriverManager.getConnection(URL, info);
```

Oracle

```
// Load the JDBC driver
```

```
Class.forName( "com.amazonaws.secretsmanager.sql.AWSSecretsManagerOracleDriver" ).newInstanc

// Retrieve the connection info from the secret using the secret ARN
String URL = "secretId";

// Populate the user property with the secret ARN to retrieve user and password from
 the secret
Properties info = new Properties( );
info.put( "user", "secretId" );

// Establish the connection
conn = DriverManager.getConnection(URL, info);
```

MSSQLServer

```
// Load the JDBC driver
Class.forName( "com.amazonaws.secretsmanager.sql.AWSSecretsManagerMSSQLServerDriver" ).newIn

// Retrieve the connection info from the secret using the secret ARN
String URL = "secretId";

// Populate the user property with the secret ARN to retrieve user and password from
 the secret
Properties info = new Properties( );
info.put( "user", "secretId" );

// Establish the connection
conn = DriverManager.getConnection(URL, info);
```

Db2

```
// Load the JDBC driver
Class.forName( "com.amazonaws.secretsmanager.sql.AWSSecretsManagerDb2Driver" ).newInstance()

// Retrieve the connection info from the secret using the secret ARN
String URL = "secretId";

// Populate the user property with the secret ARN to retrieve user and password from
 the secret
Properties info = new Properties( );
info.put( "user", "secretId" );

// Establish the connection
```

```
conn = DriverManager.getConnection(URL, info);
```

Redshift

```
// Load the JDBC driver
Class.forName( "com.amazonaws.secretsmanager.sql.AWSSecretsManagerRedshiftDriver" ).newInsta

// Retrieve the connection info from the secret using the secret ARN
String URL = "secretId";

// Populate the user property with the secret ARN to retrieve user and password from
 the secret
Properties info = new Properties( );
info.put( "user", "secretId" );

// Establish the connection
conn = DriverManager.getConnection(URL, info);
```

Buat koneksi dengan menentukan titik akhir dan port

Contoh berikut menunjukkan cara membuat koneksi ke database menggunakan kredensil secara rahasia dengan titik akhir dan port yang Anda tentukan.

Rahasia terkelola Amazon RDS tidak menyertakan titik akhir dan port database. Untuk menyambung ke database menggunakan kredensi master dalam rahasia yang dikelola oleh Amazon RDS, Anda menentukannya dalam kode Anda.

Rahasia yang direplikasi ke Wilayah lain dapat meningkatkan latensi untuk koneksi ke database regional, tetapi mereka tidak mengandung informasi koneksi yang berbeda dari rahasia sumber. Setiap replika adalah salinan rahasia sumber. Untuk menyimpan informasi koneksi regional secara rahasia, tambahkan lebih banyak key/value pasangan untuk titik akhir dan informasi port untuk Wilayah.

Setelah Anda memiliki koneksi, Anda dapat menggunakan panggilan JDBC untuk mengakses database. Untuk informasi selengkapnya, lihat JDBC Basics di situs web dokumentasi Java.

MySQL

```
// Load the JDBC driver
Class.forName( "com.amazonaws.secretsmanager.sql.AWSSecretsManagerMySQLDriver" ).newInstance
```

```
// Set the endpoint and port. You can also retrieve it from a key/value pair in the
 secret.
String URL = "jdbc-secretsmanager:mysql://example.com:3306";

// Populate the user property with the secret ARN to retrieve user and password from
 the secret
Properties info = new Properties( );
info.put( "user", "secretId" );

// Establish the connection
conn = DriverManager.getConnection(URL, info);
```

PostgreSQL

```
// Load the JDBC driver
Class.forName( "com.amazonaws.secretsmanager.sql.AWSSecretsManagerPostgreSQLDriver" ).newIns

// Set the endpoint and port. You can also retrieve it from a key/value pair in the
 secret.
String URL = "jdbc-secretsmanager:postgresql://example.com:5432/database";

// Populate the user property with the secret ARN to retrieve user and password from
 the secret
Properties info = new Properties( );
info.put( "user", "secretId" );

// Establish the connection
conn = DriverManager.getConnection(URL, info);
```

Oracle

```
// Load the JDBC driver
Class.forName( "com.amazonaws.secretsmanager.sql.AWSSecretsManagerOracleDriver" ).newInstanc

// Set the endpoint and port. You can also retrieve it from a key/value pair in the
 secret.
String URL = "jdbc-secretsmanager:oracle:thin:@example.com:1521/ORCL";

// Populate the user property with the secret ARN to retrieve user and password from
 the secret
Properties info = new Properties( );
info.put( "user", "secretId" );
```

```
// Establish the connection
conn = DriverManager.getConnection(URL, info);
```

MSSQLServer

```
// Load the JDBC driver
Class.forName( "com.amazonaws.secretsmanager.sql.AWSSecretsManagerMSSQLServerDriver" ).newIn

// Set the endpoint and port. You can also retrieve it from a key/value pair in the
 secret.
String URL = "jdbc-secretsmanager:sqlserver://example.com:1433";

// Populate the user property with the secret ARN to retrieve user and password from
 the secret
Properties info = new Properties( );
info.put( "user", "secretId" );

// Establish the connection
conn = DriverManager.getConnection(URL, info);
```

Db2

```
// Load the JDBC driver
Class.forName( "com.amazonaws.com.amazonaws.secretsmanager.sql.AWSSecretsManagerDb2Driver" )

// Set the endpoint and port. You can also retrieve it from a key/value pair in the
 secret.
String URL = "jdbc-secretsmanager:db2://example.com:50000";

// Populate the user property with the secret ARN to retrieve user and password from
 the secret
Properties info = new Properties( );
info.put( "user", "secretId" );

// Establish the connection
conn = DriverManager.getConnection(URL, info);
```

Redshift

```
// Load the JDBC driver
Class.forName( "com.amazonaws.com.amazonaws.secretsmanager.sql.AWSSecretsManagerRedshiftDriv
```

```
// Set the endpoint and port. You can also retrieve it from a key/value pair in the
 secret.
String URL = "jdbc-secretsmanager:redshift://example.com:5439";

// Populate the user property with the secret ARN to retrieve user and password from
 the secret
Properties info = new Properties( );
info.put( "user", "secretId" );

// Establish the connection
conn = DriverManager.getConnection(URL, info);
```

Gunakan penyatuan koneksi c3p0 untuk membuat koneksi

Contoh berikut menunjukkan cara membuat kolam koneksi dengan `c3p0.properties` file yang menggunakan driver untuk mengambil kredensi dan informasi koneksi dari rahasia. Untuk `user` dan`jdbcUrl`, masukkan ID rahasia untuk mengkonfigurasi kumpulan koneksi. Kemudian Anda dapat mengambil koneksi dari kolam dan menggunakannya sebagai koneksi database lainnya. Untuk informasi selengkapnya, lihat JDBC Basics di situs web dokumentasi Java.

Untuk informasi lebih lanjut tentang c3p0, lihat c3p0 di situs web Machinery For Change.

MySQL

```
c3p0.user=secretId
c3p0.driverClass=com.amazonaws.secretsmanager.sql.AWSSecretsManagerMySQLDriver
c3p0.jdbcUrl=secretId
```

PostgreSQL

```
c3p0.user=secretId
c3p0.driverClass=com.amazonaws.secretsmanager.sql.AWSSecretsManagerPostgreSQLDriver
c3p0.jdbcUrl=secretId
```

Oracle

```
c3p0.user=secretId
c3p0.driverClass=com.amazonaws.secretsmanager.sql.AWSSecretsManagerOracleDriver
c3p0.jdbcUrl=secretId
```

MSSQLServer

```
c3p0.user=secretId
c3p0.driverClass=com.amazonaws.secretsmanager.sql.AWSSecretsManagerMSSQLServerDriver
c3p0.jdbcUrl=secretId
```

Db2

```
c3p0.user=secretId
c3p0.driverClass=com.amazonaws.secretsmanager.sql.AWSSecretsManagerDb2Driver
c3p0.jdbcUrl=secretId
```

Redshift

```
c3p0.user=secretId
c3p0.driverClass=com.amazonaws.secretsmanager.sql.AWSSecretsManagerRedshiftDriver
c3p0.jdbcUrl=secretId
```

Gunakan penyatuan koneksi c3p0 untuk membuat koneksi dengan menentukan titik akhir dan port

Contoh berikut menunjukkan cara membuat kumpulan koneksi dengan `c3p0.properties` file yang menggunakan driver untuk mengambil kredensil secara rahasia dengan titik akhir dan port yang Anda tentukan. Kemudian Anda dapat mengambil koneksi dari kolam dan menggunakannya sebagai koneksi database lainnya. Untuk informasi selengkapnya, lihat JDBC Basics di situs web dokumentasi Java.

Rahasia terkelola Amazon RDS tidak menyertakan titik akhir dan port database. Untuk menyambung ke database menggunakan kredensi master dalam rahasia yang dikelola oleh Amazon RDS, Anda menentukannya dalam kode Anda.

Rahasia yang direplikasi ke Wilayah lain dapat meningkatkan latensi untuk koneksi ke database regional, tetapi mereka tidak mengandung informasi koneksi yang berbeda dari rahasia sumber. Setiap replika adalah salinan rahasia sumber. Untuk menyimpan informasi koneksi regional secara rahasia, tambahkan lebih banyak key/value pasangan untuk titik akhir dan informasi port untuk Wilayah.

MySQL

```
c3p0.user=secretId
c3p0.driverClass=com.amazonaws.secretsmanager.sql.AWSSecretsManagerMySQLDriver
c3p0.jdbcUrl=jdbc-secretsmanager:mysql://example.com:3306
```

PostgreSQL

```
c3p0.user=secretId
c3p0.driverClass=com.amazonaws.secretsmanager.sql.AWSSecretsManagerPostgreSQLDriver
c3p0.jdbcUrl=jdbc-secretsmanager:postgresql://example.com:5432/database
```

Oracle

```
c3p0.user=secretId
c3p0.driverClass=com.amazonaws.secretsmanager.sql.AWSSecretsManagerOracleDriver
c3p0.jdbcUrl=jdbc-secretsmanager:oracle:thin:@example.com:1521/ORCL
```

MSSQLServer

```
c3p0.user=secretId
c3p0.driverClass=com.amazonaws.secretsmanager.sql.AWSSecretsManagerMSSQLServerDriver
c3p0.jdbcUrl=jdbc-secretsmanager:sqlserver://example.com:1433
```

Db2

```
c3p0.user=secretId
c3p0.driverClass=com.amazonaws.secretsmanager.sql.AWSSecretsManagerDb2Driver
c3p0.jdbcUrl=jdbc-secretsmanager:db2://example.com:50000
```

Redshift

```
c3p0.user=secretId
c3p0.driverClass=com.amazonaws.secretsmanager.sql.AWSSecretsManagerRedshiftDriver
c3p0.jdbcUrl=jdbc-secretsmanager:redshift://example.com:5439
```

Dapatkan nilai rahasia Secrets Manager menggunakan Java AWS SDK

Dalam aplikasi, Anda dapat mengambil rahasia Anda dengan menelepon `GetSecretValue` atau `BatchGetSecretValue` di salah satu AWS SDK. Namun, kami menyarankan Anda menyimpan

nilai rahasia Anda dengan menggunakan caching sisi klien. Rahasia caching meningkatkan kecepatan dan mengurangi biaya Anda.

- Jika Anda menyimpan kredensyal database dalam rahasia, gunakan driver koneksi SQL Secrets Manager untuk terhubung ke database menggunakan kredensyal dalam rahasia.
- Untuk jenis rahasia lainnya, gunakan komponen Java-based caching Secrets Manager atau panggil SDK secara langsung dengan `GetSecretValue`atau. `BatchGetSecretValue`

Contoh kode berikut menunjukkan cara menggunakan`GetSecretValue`.

Izin yang diperlukan: `secretsmanager:GetSecretValue`

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.secretsmanager.SecretsManagerClient;
import software.amazon.awssdk.services.secretsmanager.model.GetSecretValueRequest;
import software.amazon.awssdk.services.secretsmanager.model.GetSecretValueResponse;
import software.amazon.awssdk.services.secretsmanager.model.SecretsManagerException;

/**
 * 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
 *
 * We recommend that you cache your secret values by using client-side caching.
 *
 * Caching secrets improves speed and reduces your costs. For more information,
 * see the following documentation topic:
 *
 * https://docs.aws.amazon.com/secretsmanager/latest/userguide/retrieving-secrets.html
 */
public class GetSecretValue {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <secretName>\s

                Where:
```

```
                    secretName - The name of the secret (for example, tutorials/
MyFirstSecret).\s
                """;

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

        String secretName = args[0];
        Region region = Region.US_EAST_1;
        SecretsManagerClient secretsClient = SecretsManagerClient.builder()
                .region(region)
                .build();

        getValue(secretsClient, secretName);
        secretsClient.close();
    }

    public static void getValue(SecretsManagerClient secretsClient, String secretName)
 {
        try {
            GetSecretValueRequest valueRequest = GetSecretValueRequest.builder()
                    .secretId(secretName)
                    .build();

            GetSecretValueResponse valueResponse =
 secretsClient.getSecretValue(valueRequest);
            String secret = valueResponse.secretString();
            System.out.println(secret);

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

Dapatkan nilai rahasia Secrets Manager menggunakan Python

Dalam aplikasi, Anda dapat mengambil rahasia Anda dengan menelepon `GetSecretValue` atau `BatchGetSecretValue` di salah satu AWS SDK. Namun, kami menyarankan Anda menyimpan

nilai rahasia Anda dengan menggunakan caching sisi klien. Rahasia caching meningkatkan kecepatan dan mengurangi biaya Anda.

Topik

- Dapatkan nilai rahasia Secrets Manager menggunakan Python dengan caching sisi klien
- Dapatkan nilai rahasia Secrets Manager menggunakan Python AWS SDK
- Dapatkan sejumlah nilai rahasia Secrets Manager menggunakan Python AWS SDK

Dapatkan nilai rahasia Secrets Manager menggunakan Python dengan caching sisi klien

Saat Anda mengambil rahasia, Anda dapat menggunakan komponen Python-based caching Secrets Manager untuk men-cache untuk digunakan di masa mendatang. Mengambil rahasia yang di-cache lebih cepat daripada mengambilnya dari Secrets Manager. Karena ada biaya untuk memanggil Secrets Manager API, menggunakan cache dapat mengurangi biaya Anda. Untuk semua cara Anda dapat mengambil rahasia, lihatDapatkan rahasia.

Kebijakan cache adalah Least Recently Used (LRU), jadi ketika cache harus membuang rahasia, ia membuang rahasia yang paling jarang digunakan. Secara default, cache menyegarkan rahasia setiap jam. Anda dapat mengonfigurasi seberapa sering rahasia disegarkan dalam cache, dan Anda dapat menghubungkan ke pengambilan rahasia untuk menambahkan lebih banyak fungsionalitas.

Cache tidak memaksa pengumpulan sampah setelah referensi cache dibebaskan. Implementasi cache tidak termasuk pembatalan cache. Implementasi cache difokuskan di sekitar cache itu sendiri, dan tidak dikeraskan atau difokuskan keamanan. Jika Anda memerlukan keamanan tambahan seperti mengenkripsi item dalam cache, gunakan antarmuka dan metode abstrak yang disediakan.

Untuk menggunakan komponen, Anda harus memiliki yang berikut:

- Python 3.6 atau yang lebih baru.
- botocore 1.12 atau lebih tinggi. Lihat AWS SDK untuk Python dan Botocore.
- setuptools_scm 3.2 atau lebih tinggi. Lihat https://pypi.org/project/setuptools-scm/.

Untuk mengunduh kode sumber, lihat Secrets Manager Python-based caching komponen klien di GitHub.

Untuk menginstal komponen, gunakan perintah berikut.

```
$ pip install aws-secretsmanager-caching
```

Izin yang diperlukan:

- `secretsmanager:DescribeSecret`
- `secretsmanager:GetSecretValue`

Untuk informasi selengkapnya, lihat Referensi izin.

Referensi

- SecretCache
- SecretCacheConfig
- SecretCacheHook
- @InjectSecretString
- @InjectKeywordedSecretString

Example Ambil rahasia

Contoh berikut menunjukkan bagaimana untuk mendapatkan nilai rahasia untuk rahasia bernama*mysecret*.

```
import botocore
import botocore.session
from aws_secretsmanager_caching import SecretCache, SecretCacheConfig

client = botocore.session.get_session().create_client('secretsmanager')
cache_config = SecretCacheConfig()
cache = SecretCache( config = cache_config, client = client)

secret = cache.get_secret_string('mysecret')
```

SecretCache

Cache dalam memori untuk rahasia yang diambil dari Secrets Manager. Anda menggunakan the section called “get_secret_string” atau the section called “get_secret_binary” untuk mengambil rahasia dari cache. Anda dapat mengkonfigurasi pengaturan cache dengan meneruskan the section called “SecretCacheConfig” objek di konstruktor.

Untuk informasi selengkapnya, termasuk contoh, lihatthe section called “Python dengan caching sisi klien”.

```
cache = SecretCache(
    config = the section called “SecretCacheConfig”,
    client = client
)
```

Ini adalah metode yang tersedia:

- get_secret_string
- get_secret_binary

get_secret_string

Mengambil nilai string rahasia.

Sintaksis Permintaan

```
response = cache.get_secret_string(
    secret_id='string',
    version_stage='string' )
```

Parameter

- `secret_id`(string): [Diperlukan] Nama atau ARN rahasia.
- `version_stage`(string): Versi rahasia yang ingin Anda ambil. Untuk informasi selengkapnya, lihat versi rahasia. Defaultnya adalah 'AWSCURRENT'.

Jenis pengembalian

string

get_secret_binary

Mengambil nilai biner rahasia.

Sintaksis Permintaan

```
response = cache.get_secret_binary(
    secret_id='string',
```

```
    version_stage='string'
)
```

Parameter

- `secret_id`(string): [Diperlukan] Nama atau ARN rahasia.
- `version_stage`(string): Versi rahasia yang ingin Anda ambil. Untuk informasi selengkapnya, lihat versi rahasia. Defaultnya adalah 'AWSCURRENT'.

Jenis pengembalian

string yang dikodekan base64

SecretCacheConfig

Opsi konfigurasi cache untuk ukuran cache maks dan Time to Live (TTL) untuk rahasia cache. the section called “SecretCache”

Parameter

`max_cache_size`(int)

Ukuran cache maksimum. Defaultnya adalah `1024` rahasia.

`exception_retry_delay_base`(int)

Jumlah detik untuk menunggu setelah pengecualian ditemui sebelum mencoba kembali permintaan. Nilai default-nya `1`.

`exception_retry_growth_factor`(int) pur

Faktor pertumbuhan yang digunakan untuk menghitung waktu tunggu antara percobaan ulang permintaan yang gagal. Nilai default-nya `2`.

`exception_retry_delay_max`(int)

Jumlah maksimum waktu dalam hitungan detik untuk menunggu di antara permintaan yang gagal. Nilai default-nya `3600`.

`default_version_stage`(str)

Versi rahasia yang ingin Anda cache. Untuk informasi selengkapnya, lihat Versi rahasia. Nilai default-nya `'AWSCURRENT'`.

`secret_refresh_interval`(int)

Jumlah detik untuk menunggu antara menyegarkan informasi rahasia yang di-cache. Nilai default-nya `3600`.

`secret_cache_hook` (SecretCacheHook)

Implementasi kelas `SecretCacheHook` abstrak. Nilai default-nya adalah `None`.

SecretCacheHook

Antarmuka untuk menghubungkan the section called “SecretCache” ke dalam untuk melakukan tindakan pada rahasia yang disimpan dalam cache.

Ini adalah metode yang tersedia:

- menempatkan
- memperoleh

menempatkan

Mempersiapkan objek untuk disimpan dalam cache.

Sintaksis Permintaan

```
response = hook.put(
    obj='secret_object'
)
```

Parameter

- `obj`(objek) -- [Diperlukan] Rahasia atau objek yang berisi rahasia.

Jenis pengembalian

object

memperoleh

Mendapatkan objek dari objek yang di-cache.

Sintaksis Permintaan

```
response = hook.get(
    obj='secret_object'
)
```

Parameter

- `obj`(objek): [Wajib] Rahasia atau objek yang berisi rahasia.

Jenis pengembalian

object

@InjectSecretString

Dekorator ini mengharapkan string ID rahasia dan the section called “SecretCache” sebagai argumen pertama dan kedua. Dekorator mengembalikan nilai string rahasia. Rahasianya harus berisi string.

```
from aws_secretsmanager_caching import SecretCache
from aws_secretsmanager_caching import InjectKeywordedSecretString,
 InjectSecretString

cache = SecretCache()

@InjectSecretString ( 'mysecret' ,  cache )
def function_to_be_decorated( arg1,  arg2,  arg3):
```

@InjectKeywordedSecretString

Dekorator ini mengharapkan string ID rahasia dan the section called “SecretCache” sebagai argumen pertama dan kedua. Argumen yang tersisa memetakan parameter dari fungsi yang dibungkus ke kunci JSON dalam rahasia. Rahasianya harus berisi string dalam struktur JSON.

Untuk rahasia yang berisi JSON ini:

```
{
  "username": "saanvi",
  "password": "EXAMPLE-PASSWORD"
}
```

Contoh berikut menunjukkan cara mengekstrak nilai JSON untuk `username` dan `password` dari rahasia.

```
from aws_secretsmanager_caching import SecretCache
  from aws_secretsmanager_caching import InjectKeywordedSecretString,
 InjectSecretString

  cache = SecretCache()

  @InjectKeywordedSecretString ( secret_id = 'mysecret' ,  cache = cache ,
 func_username = 'username' ,  func_password = 'password' )
  def function_to_be_decorated( func_username,  func_password):
       print( 'Do something with the func_username and func_password parameters')
```

Dapatkan nilai rahasia Secrets Manager menggunakan Python AWS SDK

Dalam aplikasi, Anda dapat mengambil rahasia Anda dengan menelepon `GetSecretValue` atau `BatchGetSecretValue` di salah satu AWS SDK. Namun, kami menyarankan Anda menyimpan nilai rahasia Anda dengan menggunakan caching sisi klien. Rahasia caching meningkatkan kecepatan dan mengurangi biaya Anda.

Untuk aplikasi Python, gunakan komponen Python-based caching Secrets Manager atau panggil SDK langsung dengan atau. `get_secret_valuebatch_get_secret_value`

Contoh kode berikut menunjukkan cara menggunakan`GetSecretValue`.

Izin yang diperlukan: `secretsmanager:GetSecretValue`

```
"""
Purpose

Shows how to use the AWS SDK for Python (Boto3) with AWS
Secrets Manager to get a specific of secrets that match a
specified name
"""
import boto3
import logging

from get_secret_value import GetSecretWrapper

# Configure logging
logging.basicConfig(level=logging.INFO)
```

```
def run_scenario(secret_name):
    """
    Retrieve a secret from AWS Secrets Manager.

    :param secret_name: Name of the secret to retrieve.
    :type secret_name: str
    """
    try:
        # Validate secret_name
        if not secret_name:
            raise ValueError("Secret name must be provided.")
        # Retrieve the secret by name
        client = boto3.client("secretsmanager")
        wrapper = GetSecretWrapper(client)
        secret = wrapper.get_secret(secret_name)
        # Note: Secrets should not be logged.
        return secret
    except Exception as e:
        logging.error(f"Error retrieving secret: {e}")
        raise

class GetSecretWrapper:
    def __init__(self, secretsmanager_client):
        self.client = secretsmanager_client


    def get_secret(self, secret_name):
        """
        Retrieve individual secrets from AWS Secrets Manager using the get_secret_value
 API.
        This function assumes the stack mentioned in the source code README has been
 successfully deployed.
        This stack includes 7 secrets, all of which have names beginning with
 "mySecret".

        :param secret_name: The name of the secret fetched.
        :type secret_name: str
        """
        try:
            get_secret_value_response = self.client.get_secret_value(
                SecretId=secret_name
            )
```

```
            logging.info("Secret retrieved successfully.")
            return get_secret_value_response["SecretString"]
        except self.client.exceptions.ResourceNotFoundException:
            msg = f"The requested secret {secret_name} was not found."
            logger.info(msg)
            return msg
        except Exception as e:
            logger.error(f"An unknown error occurred: {str(e)}.")
            raise
```

Dapatkan sejumlah nilai rahasia Secrets Manager menggunakan Python AWS SDK

Contoh kode berikut menunjukkan cara mendapatkan sejumlah nilai rahasia Secrets Manager.

Izin yang diperlukan:

- `secretsmanager:BatchGetSecretValue`
- `secretsmanager:GetSecretValue`izin untuk setiap rahasia yang ingin Anda ambil.
- Jika Anda menggunakan filter, Anda juga harus memilikinya`secretsmanager:ListSecrets`.

Untuk contoh kebijakan izin, lihatthe section called “Contoh: Izin untuk mengambil sekelompok nilai rahasia dalam batch”.

> **Important**
>
> Jika Anda memiliki kebijakan VPCE yang menolak izin untuk mengambil rahasia individu dalam grup yang Anda ambil, tidak `BatchGetSecretValue` akan mengembalikan nilai rahasia apa pun, dan itu akan mengembalikan kesalahan.

```
class BatchGetSecretsWrapper:
    def __init__(self, secretsmanager_client):
        self.client = secretsmanager_client


    def batch_get_secrets(self, filter_name):
```

```
        """
        Retrieve multiple secrets from AWS Secrets Manager using the
batch_get_secret_value API.
        This function assumes the stack mentioned in the source code README has been
successfully deployed.
        This stack includes 7 secrets, all of which have names beginning with
"mySecret".

        :param filter_name: The full or partial name of secrets to be fetched.
        :type filter_name: str
        """
        try:
            secrets = []
            response = self.client.batch_get_secret_value(
                Filters=[{"Key": "name", "Values": [f"{filter_name}"]}]
            )
            for secret in response["SecretValues"]:
                secrets.append(json.loads(secret["SecretString"]))
            if secrets:
                logger.info("Secrets retrieved successfully.")
            else:
                logger.info("Zero secrets returned without error.")
            return secrets
        except self.client.exceptions.ResourceNotFoundException:
            msg = f"One or more requested secrets were not found with filter:
{filter_name}"
            logger.info(msg)
            return msg
        except Exception as e:
            logger.error(f"An unknown error occurred:\n{str(e)}.")
            raise
```

Dapatkan nilai rahasia Secrets Manager menggunakan .NET

Dalam aplikasi, Anda dapat mengambil rahasia Anda dengan menelepon `GetSecretValue` atau `BatchGetSecretValue` di salah satu AWS SDK. Namun, kami menyarankan Anda menyimpan nilai rahasia Anda dengan menggunakan caching sisi klien. Rahasia caching meningkatkan kecepatan dan mengurangi biaya Anda.

Topik

- Dapatkan nilai rahasia Secrets Manager menggunakan.NET dengan caching sisi klien
- Dapatkan nilai rahasia Secrets Manager menggunakan SDK untuk .NET

Dapatkan nilai rahasia Secrets Manager menggunakan.NET dengan caching sisi klien

Saat Anda mengambil rahasia, Anda dapat menggunakan Secrets Manager. NET-based komponen caching untuk men-cache untuk digunakan di masa mendatang. Mengambil rahasia yang di-cache lebih cepat daripada mengambilnya dari Secrets Manager. Karena ada biaya untuk memanggil Secrets Manager API, menggunakan cache dapat mengurangi biaya Anda. Untuk semua cara Anda dapat mengambil rahasia, lihatDapatkan rahasia.

Kebijakan cache adalah Least Recently Used (LRU), jadi ketika cache harus membuang rahasia, ia membuang rahasia yang paling jarang digunakan. Secara default, cache menyegarkan rahasia setiap jam. Anda dapat mengonfigurasi seberapa sering rahasia disegarkan dalam cache, dan Anda dapat menghubungkan ke pengambilan rahasia untuk menambahkan lebih banyak fungsionalitas.

Cache tidak memaksa pengumpulan sampah setelah referensi cache dibebaskan. Implementasi cache tidak termasuk pembatalan cache. Implementasi cache difokuskan di sekitar cache itu sendiri, dan tidak dikeraskan atau difokuskan keamanan. Jika Anda memerlukan keamanan tambahan seperti mengenkripsi item dalam cache, gunakan antarmuka dan metode abstrak yang disediakan.

Untuk menggunakan komponen, Anda harus memiliki yang berikut:

- .NET Framework 4.6.2 atau lebih tinggi, atau .NET Standard 2.0 atau lebih tinggi. Lihat Unduh .NET di situs web Microsoft .NET.
- AWS SDK for .NET. Lihat the section called “AWS SDK”.

Untuk mengunduh kode sumber, lihat Caching client untuk.NET di GitHub.

Untuk menggunakan cache, pertama buat instance, lalu ambil rahasia Anda dengan menggunakan atau. `GetSecretString` `GetSecretBinary` Pada pengambilan berturut-turut, cache mengembalikan salinan rahasia yang di-cache.

Untuk mendapatkan paket caching

- Lakukan salah satu tindakan berikut:
  - Jalankan perintah.NET CLI berikut di direktori proyek Anda.

```
dotnet add package AWSSDK.SecretsManager.Caching --version 1.0.6
```

- Tambahkan referensi paket berikut ke `.csproj` file Anda.

```
<ItemGroup>
    <PackageReference Include="AWSSDK.SecretsManager.Caching" Version="1.0.6" /
>
</ItemGroup>
```

Izin yang diperlukan:

- `secretsmanager:DescribeSecret`
- `secretsmanager:GetSecretValue`

Untuk informasi selengkapnya, lihat Referensi izin.

Referensi

- SecretsManagerCache
- SecretCacheConfiguration
- ISecretCacheHook

Example Ambil rahasia

Contoh kode berikut menunjukkan metode yang mengambil rahasia bernama*MySecret*.

```
using Amazon.SecretsManager.Extensions.Caching;

namespace LambdaExample
{
    public class CachingExample
    {
        private const string MySecretName ="MySecret";

        private SecretsManagerCache cache = new SecretsManagerCache();

        public async Task<Response>  FunctionHandlerAsync(string input, ILambdaContext
 context)
        {
```

```
            string MySecret = await cache.GetSecretString(MySecretName);

            // Use the secret, return success

        }
    }
}
```

Example Konfigurasikan durasi penyegaran cache time to live (TTL)

Contoh kode berikut menunjukkan metode yang mengambil rahasia bernama *MySecret* dan menetapkan durasi penyegaran cache TTL menjadi 24 jam.

```
using Amazon.SecretsManager.Extensions.Caching;

namespace LambdaExample
{
    public class CachingExample
    {
        private const string MySecretName = "MySecret";

        private static SecretCacheConfiguration cacheConfiguration = new
 SecretCacheConfiguration
        {
            CacheItemTTL = 86400000
        };
        private SecretsManagerCache cache = new
 SecretsManagerCache(cacheConfiguration);
        public async Task<Response> FunctionHandlerAsync(string input, ILambdaContext
 context)
        {
            string mySecret = await cache.GetSecretString(MySecretName);

            // Use the secret, return success
        }
    }
}
```

SecretsManagerCache

Cache dalam memori untuk rahasia yang diminta dari Secrets Manager. Anda menggunakan the section called “GetSecretString” atau the section called “GetSecretBinary” untuk mengambil rahasia

dari cache. Anda dapat mengkonfigurasi pengaturan cache dengan meneruskan the section called "SecretCacheConfiguration" objek di konstruktor.

Untuk informasi selengkapnya, termasuk contoh, lihatthe section called ".NET dengan caching sisi klien".

Konstruktor

```
public SecretsManagerCache()
```

Konstruktor default untuk `SecretsManagerCache` objek.

```
public SecretsManagerCache(IAmazonSecretsManager secretsManager)
```

Membangun cache baru menggunakan klien Secrets Manager yang dibuat menggunakan yang disediakan AmazonSecretsManagerClient. Gunakan konstruktor ini untuk menyesuaikan klien Secrets Manager, misalnya untuk menggunakan wilayah atau titik akhir tertentu.

Parameter

Rahasia Manajer

AmazonSecretsManagerClientUntuk mengambil rahasia dari.

```
public SecretsManagerCache(SecretCacheConfiguration config)
```

Membangun cache rahasia baru menggunakan yang disediakanthe section called "SecretCacheConfiguration". Gunakan konstruktor ini untuk mengkonfigurasi cache, misalnya jumlah rahasia untuk cache dan seberapa sering itu menyegarkan.

Parameter

config

A the section called "SecretCacheConfiguration" yang berisi informasi konfigurasi untuk cache.

```
public SecretsManagerCache(IAmazonSecretsManager secretsManager,
SecretCacheConfiguration config)
```

Membangun cache baru menggunakan klien Secrets Manager yang dibuat menggunakan yang disediakan AmazonSecretsManagerClientdan file. the section called "SecretCacheConfiguration" Gunakan konstruktor ini untuk menyesuaikan klien Secrets Manager, misalnya untuk menggunakan wilayah atau titik akhir tertentu serta mengkonfigurasi cache, misalnya jumlah rahasia untuk cache dan seberapa sering itu menyegarkan.

Parameter

Rahasia Manajer

AmazonSecretsManagerClientUntuk mengambil rahasia dari.

config

A the section called “SecretCacheConfiguration” yang berisi informasi konfigurasi untuk cache.

Metode

GetSecretString

```
public async Task<String> GetSecretString(String secretId)
```

Mengambil rahasia string dari Secrets Manager.

Parameter

secretId

ARN atau nama rahasia untuk diambil.

GetSecretBinary

```
public async Task<byte[]> GetSecretBinary(String secretId)
```

Mengambil rahasia biner dari Secrets Manager.

Parameter

secretId

ARN atau nama rahasia untuk diambil.

RefreshNowAsync

```
public async Task<bool> RefreshNowAsync(String secretId)
```

Meminta nilai rahasia dari Secrets Manager dan memperbarui cache dengan perubahan apa pun. Jika tidak ada entri cache yang ada, buat yang baru. Kembali `true` jika refresh berhasil.

Parameter

secretId

ARN atau nama rahasia untuk diambil.

GetCachedSecret

```
public SecretCacheItem GetCachedSecret(string secretId)
```

Mengembalikan entri cache untuk rahasia tertentu jika ada dalam cache. Jika tidak, mengambil rahasia dari Secrets Manager dan membuat entri cache baru.

Parameter

secretId

ARN atau nama rahasia untuk diambil.

SecretCacheConfiguration

Opsi konfigurasi cache untukthe section called “SecretsManagerCache”, seperti ukuran cache maksimum dan Time to Live (TTL) untuk rahasia cache.

Sifat-sifat

CacheItemTTL

```
public uint CacheItemTTL { get; set; }
```

TTL item cache dalam milidetik. Defaultnya adalah `3600000` ms atau 1 jam. Maksimum adalah `4294967295` ms, yaitu sekitar 49,7 hari.

MaxCacheSize

```
public ushort MaxCacheSize { get; set; }
```

Ukuran cache maksimum. Defaultnya adalah 1024 rahasia. Maksimum adalah 65.535.

VersionStage

```
public string VersionStage { get; set; }
```

Versi rahasia yang ingin Anda cache. Untuk informasi selengkapnya, lihat Versi rahasia. Nilai default-nya `"AWSCURRENT"`.

Klien

```
public IAmazonSecretsManager Client { get; set; }
```

AmazonSecretsManagerClientUntuk mengambil rahasia dari. Jika yanull, cache membuat instance klien baru. Nilai default-nya `null`.

CacheHook

```
public ISecretCacheHook CacheHook { get; set; }
```

Athe section called “ISecretCacheHook”.

ISecretCacheHook

Antarmuka untuk menghubungkan the section called “SecretsManagerCache” ke dalam untuk melakukan tindakan pada rahasia yang disimpan dalam cache.

Metode

Masukan

```
object Put(object o);
```

Siapkan objek untuk disimpan dalam cache.

Mengembalikan objek untuk menyimpan dalam cache.

Dapatkan

```
object Get(object cachedObject);
```

Turunkan objek dari objek yang di-cache.

Mengembalikan objek untuk kembali dari cache

Dapatkan nilai rahasia Secrets Manager menggunakan SDK untuk .NET

Dalam aplikasi, Anda dapat mengambil rahasia Anda dengan menelepon `GetSecretValue` atau `BatchGetSecretValue` di salah satu AWS SDK. Namun, kami menyarankan Anda menyimpan nilai rahasia Anda dengan menggunakan caching sisi klien. Rahasia caching meningkatkan kecepatan dan mengurangi biaya Anda.

Untuk aplikasi.NET, gunakan Secrets Manager. NET-based caching komponen atau panggil SDK langsung dengan `GetSecretValue`atau. `BatchGetSecretValue`

Contoh kode berikut menunjukkan cara menggunakan`GetSecretValue`.

Izin yang diperlukan: `secretsmanager:GetSecretValue`

```
using System;
using System.IO;
using System.Threading.Tasks;
using Amazon.SecretsManager;
using Amazon.SecretsManager.Model;

/// <summary>
/// This example uses the Amazon Web Service Secrets Manager to retrieve
/// the secret value for the provided secret name.
/// </summary>
public class GetSecretValue
{
    /// <summary>
    /// The main method initializes the necessary values and then calls
    /// the GetSecretAsync and DecodeString methods to get the decoded
    /// secret value for the secret named in secretName.
    /// </summary>
    public static async Task Main()
    {
        string secretName = "<<{{MySecretName}}>>";
        string secret;

        IAmazonSecretsManager client = new AmazonSecretsManagerClient();

        var response = await GetSecretAsync(client, secretName);

        if (response is not null)
        {
            secret = DecodeString(response);

            if (!string.IsNullOrEmpty(secret))
            {
                Console.WriteLine($"The decoded secret value is: {secret}.");
            }
            else
            {
                Console.WriteLine("No secret value was returned.");
```

```
                }
            }
        }

        /// <summary>
        /// Retrieves the secret value given the name of the secret to
        /// retrieve.
        /// </summary>
        /// <param name="client">The client object used to retrieve the secret
        /// value for the given secret name.</param>
        /// <param name="secretName">The name of the secret value to retrieve.</param>
        /// <returns>The GetSecretValueReponse object returned by
        /// GetSecretValueAsync.</returns>
        public static async Task<GetSecretValueResponse> GetSecretAsync(
            IAmazonSecretsManager client,
            string secretName)
        {
            GetSecretValueRequest request = new GetSecretValueRequest()
            {
                SecretId = secretName,
                VersionStage = "AWSCURRENT", // VersionStage defaults to AWSCURRENT if
unspecified.
            };

            GetSecretValueResponse response = null;

            // For the sake of simplicity, this example handles only the most
            // general SecretsManager exception.
            try
            {
                response = await client.GetSecretValueAsync(request);
            }
            catch (AmazonSecretsManagerException e)
            {
                Console.WriteLine($"Error: {e.Message}");
            }

            return response;
        }

        /// <summary>
        /// Decodes the secret returned by the call to GetSecretValueAsync and
        /// returns it to the calling program.
        /// </summary>
```

```
        /// <param name="response">A GetSecretValueResponse object containing
        /// the requested secret value returned by GetSecretValueAsync.</param>
        /// <returns>A string representing the decoded secret value.</returns>
        public static string DecodeString(GetSecretValueResponse response)
        {
            // Decrypts secret using the associated AWS Key Management Service
            // Customer Master Key (CMK.) Depending on whether the secret is a
            // string or binary value, one of these fields will be populated.
            if (response.SecretString is not null)
            {
                var secret = response.SecretString;
                return secret;
            }
            else if (response.SecretBinary is not null)
            {
                var memoryStream = response.SecretBinary;
                StreamReader reader = new StreamReader(memoryStream);
                string decodedBinarySecret =
System.Text.Encoding.UTF8.GetString(Convert.FromBase64String(reader.ReadToEnd()));
                return decodedBinarySecret;
            }
            else
            {
                return string.Empty;
            }
        }
    }
```

Dapatkan nilai rahasia Secrets Manager menggunakan Go

Dalam aplikasi, Anda dapat mengambil rahasia Anda dengan menelepon `GetSecretValue` atau `BatchGetSecretValue` di salah satu AWS SDK. Namun, kami menyarankan Anda menyimpan nilai rahasia Anda dengan menggunakan caching sisi klien. Rahasia caching meningkatkan kecepatan dan mengurangi biaya Anda.

Topik

- Dapatkan nilai rahasia Secrets Manager menggunakan Go dengan caching sisi klien
- Dapatkan nilai rahasia Secrets Manager menggunakan Go AWS SDK

Dapatkan nilai rahasia Secrets Manager menggunakan Go dengan caching sisi klien

Saat Anda mengambil rahasia, Anda dapat menggunakan komponen Go-based caching Secrets Manager untuk men-cache untuk digunakan di masa mendatang. Mengambil rahasia yang di-cache lebih cepat daripada mengambilnya dari Secrets Manager. Karena ada biaya untuk memanggil Secrets Manager API, menggunakan cache dapat mengurangi biaya Anda. Untuk semua cara Anda dapat mengambil rahasia, lihatDapatkan rahasia.

Kebijakan cache adalah Least Recently Used (LRU), jadi ketika cache harus membuang rahasia, ia membuang rahasia yang paling jarang digunakan. Secara default, cache menyegarkan rahasia setiap jam. Anda dapat mengonfigurasi seberapa sering rahasia disegarkan dalam cache, dan Anda dapat menghubungkan ke pengambilan rahasia untuk menambahkan lebih banyak fungsionalitas.

Cache tidak memaksa pengumpulan sampah setelah referensi cache dibebaskan. Implementasi cache tidak termasuk pembatalan cache. Implementasi cache difokuskan di sekitar cache itu sendiri, dan tidak dikeraskan atau difokuskan keamanan. Jika Anda memerlukan keamanan tambahan seperti mengenkripsi item dalam cache, gunakan antarmuka dan metode abstrak yang disediakan.

Untuk menggunakan komponen, Anda harus memiliki yang berikut:

- AWS SDK for Go. Lihat the section called “AWS SDK”.

Untuk mengunduh kode sumber, lihat Secrets Manager Go caching client di GitHub.

Untuk menyiapkan lingkungan pengembangan Go, lihat Golang Memulai di situs web Go Programming Language.

Izin yang diperlukan:

- `secretsmanager:DescribeSecret`
- `secretsmanager:GetSecretValue`

Untuk informasi selengkapnya, lihat Referensi izin.

Referensi

- jenis Cache
- jenis CacheConfig

- jenis CacheHook

Example Ambil rahasia

Contoh kode berikut menunjukkan fungsi Lambda yang mengambil rahasia.

```
package main

import (
  "github.com/aws/aws-lambda-go/lambda"
  "github.com/aws/aws-secretsmanager-caching-go/v2/secretcache"
)

var (
  secretCache, _ = secretcache.New()
)

func HandleRequest(secretId string) string {
  result, _ := secretCache.GetSecretString(secretId)

  // Use the secret, return success
}

 func main() {
  lambda. Start( HandleRequest)
}
```

jenis Cache

Cache dalam memori untuk rahasia yang diminta dari Secrets Manager. Anda menggunakan the section called “GetSecretString” atau the section called “GetSecretBinary” untuk mengambil rahasia dari cache.

Contoh berikut menunjukkan cara mengkonfigurasi pengaturan cache.

```
// Create a custom secretsmanager client
client := getCustomClient()

// Create a custom CacheConfig struct
config := secretcache. CacheConfig{
    MaxCacheSize:  secretcache.DefaultMaxCacheSize + 10,
    VersionStage:  secretcache.DefaultVersionStage,
```

```
    CacheItemTTL:  secretcache.DefaultCacheItemTTL,
}

// Instantiate the cache
cache, _ := secretcache.New(
    func( c *secretcache.Cache) {  c. CacheConfig = config },
    func( c *secretcache.Cache) {  c. Client = client },
)
```

Untuk informasi selengkapnya, termasuk contoh, lihatthe section called “Pergi dengan caching sisi klien”.

Metode

Baru

```
func New(optFns ...func(*Cache)) (*Cache, error)
```

Baru membangun cache rahasia menggunakan opsi fungsional, menggunakan default sebaliknya. Menginisialisasi SecretsManager Klien dari sesi baru. Menginisialisasi CacheConfig ke nilai default. Menginisialisasi cache LRU dengan ukuran maks default.

GetSecretString

```
func (c *Cache) GetSecretString(secretId string) (string, error)
```

GetSecretString mendapatkan nilai string rahasia dari cache untuk ID rahasia yang diberikan. Mengembalikan string rahasia dan kesalahan jika operasi gagal.

GetSecretStringWithStage

```
func (c *Cache) GetSecretStringWithStage(secretId string, versionStage
string) (string, error)
```

GetSecretStringWithStage mendapatkan nilai string rahasia dari cache untuk ID rahasia dan tahap versi yang diberikan. Mengembalikan string rahasia dan kesalahan jika operasi gagal.

GetSecretBinary

```
func (c *Cache) GetSecretBinary(secretId string) ([]byte, error) {
```

GetSecretBinary mendapatkan nilai biner rahasia dari cache untuk ID rahasia yang diberikan. Mengembalikan biner rahasia dan kesalahan jika operasi gagal.

GetSecretBinaryWithStage

```
func (c *Cache) GetSecretBinaryWithStage(secretId string, versionStage
string) ([]byte, error)
```

GetSecretBinaryWithStage mendapatkan nilai biner rahasia dari cache untuk ID rahasia dan tahap versi yang diberikan. Mengembalikan biner rahasia dan kesalahan jika operasi gagal.

jenis CacheConfig

Opsi konfigurasi cache untuk Cache, seperti ukuran cache maksimum, tahap versi default, dan Time to Live (TTL) untuk rahasia cache.

```
type CacheConfig struct {

    // The maximum cache size. The default is 1024 secrets.
    MaxCacheSize int

    // The TTL of a cache item in nanoseconds. The default is
    // 3.6e10^12 ns or 1 hour.
    CacheItemTTL int64

    // The version of secrets that you want to cache. The default
    // is "AWSCURRENT".
    VersionStage string

    // Used to hook in-memory cache updates.
    Hook CacheHook
}
```

jenis CacheHook

Antarmuka untuk menghubungkan ke Cache untuk melakukan tindakan pada rahasia yang disimpan dalam cache.

Metode

Masukan

```
Put(data interface{}) interface{}
```

Mempersiapkan objek untuk disimpan dalam cache.

Dapatkan

`Get(data interface{}) interface{}`

Mendapatkan objek dari objek yang di-cache.

Dapatkan nilai rahasia Secrets Manager menggunakan Go AWS SDK

Dalam aplikasi, Anda dapat mengambil rahasia Anda dengan menelepon `GetSecretValue` atau `BatchGetSecretValue` di salah satu AWS SDK. Namun, kami menyarankan Anda menyimpan nilai rahasia Anda dengan menggunakan caching sisi klien. Rahasia caching meningkatkan kecepatan dan mengurangi biaya Anda.

Untuk aplikasi Go, gunakan komponen Go-based caching Secrets Manager atau panggil SDK langsung dengan `GetSecretValue`atau. `BatchGetSecretValue`

Contoh kode berikut menunjukkan cara mendapatkan nilai rahasia Secrets Manager.

Izin yang diperlukan: `secretsmanager:GetSecretValue`

```
 // Use this code snippet in your app.
 // If you need more information about configurations or implementing the sample code,
visit the AWS docs:
 // https://aws.github.io/aws-sdk-go-v2/docs/getting-started/

 import (
   "context"
   "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/secretsmanager"
 )

 func main() {
   secretName := "<<{{MySecretName}}>>"
   region := "<<{{MyRegionName}}>>"

   config, err := config.LoadDefaultConfig(context.TODO(), config.WithRegion(region))
   if err != nil {
     log.Fatal(err)
   }
```

```
    // Create Secrets Manager client
    svc := secretsmanager.NewFromConfig(config)

    input := &secretsmanager.GetSecretValueInput{
      SecretId:     aws.String(secretName),
      VersionStage: aws.String("AWSCURRENT"), // VersionStage defaults to AWSCURRENT if
 unspecified
    }

    result, err := svc.GetSecretValue(context.TODO(), input)
    if err != nil {
      // For a list of exceptions thrown, see
      // https://<<{{DocsDomain}}>>/secretsmanager/latest/apireference/
API_GetSecretValue.html
      log.Fatal(err.Error())
    }

    // Decrypts secret using the associated KMS key.
    var secretString string = *result.SecretString

    // Your code goes here.
  }
```

Dapatkan nilai rahasia Secrets Manager menggunakan Rust

Dalam aplikasi, Anda dapat mengambil rahasia Anda dengan menelepon `GetSecretValue` atau `BatchGetSecretValue` di salah satu AWS SDK. Namun, kami menyarankan Anda menyimpan nilai rahasia Anda dengan menggunakan caching sisi klien. Rahasia caching meningkatkan kecepatan dan mengurangi biaya Anda.

Topik

- Dapatkan nilai rahasia Secrets Manager menggunakan Rust dengan caching sisi klien
- Dapatkan nilai rahasia Secrets Manager menggunakan Rust AWS SDK

Dapatkan nilai rahasia Secrets Manager menggunakan Rust dengan caching sisi klien

Saat Anda mengambil rahasia, Anda dapat menggunakan komponen Rust-based caching Secrets Manager untuk men-cache untuk digunakan di masa mendatang. Mengambil rahasia yang di-cache

lebih cepat daripada mengambilnya dari Secrets Manager. Karena ada biaya untuk memanggil Secrets Manager API, menggunakan cache dapat mengurangi biaya Anda. Untuk semua cara Anda dapat mengambil rahasia, lihatDapatkan rahasia.

Kebijakan cache adalah First In First Out (FIFO), jadi ketika cache harus membuang rahasia, ia membuang rahasia tertua. Secara default, cache menyegarkan rahasia setiap jam. Anda dapat mengonfigurasi yang berikut:

- `max_size`— Jumlah maksimum rahasia cache yang harus disimpan sebelum mengusir rahasia yang belum diakses baru-baru ini.
- `ttl`— Durasi item yang di-cache dianggap valid sebelum memerlukan penyegaran keadaan rahasia.

Implementasi cache tidak termasuk pembatalan cache. Implementasi cache difokuskan di sekitar cache itu sendiri, dan tidak dikeraskan atau difokuskan keamanan. Jika Anda memerlukan keamanan tambahan seperti mengenkripsi item dalam cache, gunakan sifat yang disediakan untuk memodifikasi cache.

Untuk menggunakan komponen, Anda harus memiliki lingkungan pengembangan Rust 2021 dengan`tokio`. Untuk informasi selengkapnya, lihat Memulai situs web Rust Programming Language.

Untuk mengunduh kode sumber, lihat Secrets Manager Rust-based caching komponen klien di GitHub.

Untuk menginstal komponen caching, gunakan perintah berikut.

```
cargo add aws_secretsmanager_caching
```

Izin yang diperlukan:

- `secretsmanager:DescribeSecret`
- `secretsmanager:GetSecretValue`

Untuk informasi selengkapnya, lihat Referensi izin.

Example Ambil rahasia

Contoh berikut menunjukkan bagaimana untuk mendapatkan nilai rahasia untuk rahasia bernama*`MyTest`*.

```
use aws_secretsmanager_caching::SecretsManagerCachingClient;
use std::num::NonZeroUsize;
use std::time::Duration;

let client = match SecretsManagerCachingClient::default(
    NonZeroUsize::new(10).unwrap(),
    Duration::from_secs(60),
)
.await
{
    Ok(c) => c,
    Err(_) => panic!("Handle this error"),
};

let secret_string = match client.get_secret_value("MyTest", None, None).await {
    Ok(s) => s.secret_string.unwrap(),
    Err(_) => panic!("Handle this error"),
};

// Your code here
```

Example Membuat Instantiasi Cache dengan konfigurasi khusus dan klien khusus

Contoh berikut menunjukkan cara mengkonfigurasi cache dan kemudian mendapatkan nilai rahasia untuk rahasia bernama*MyTest*.

```
let config = aws_config::load_defaults(BehaviorVersion::latest())
    .await
    .into_builder()
    .region(Region::from_static("us-west-2"))
    .build();

let asm_builder = aws_sdk_secretsmanager::config::Builder::from(&config);

let client = match SecretsManagerCachingClient::from_builder(
    asm_builder,
    NonZeroUsize::new(10).unwrap(),
    Duration::from_secs(60),
)
.await
{
    Ok(c) => c,
    Err(_) => panic!("Handle this error"),
```

```
};

let secret_string = client
    .get_secret_value("MyTest", None, None)
    .await
    {
        Ok(c) => c.secret_string.unwrap(),
        Err(_) => panic!("Handle this error"),
    };

// Your code here
```

Dapatkan nilai rahasia Secrets Manager menggunakan Rust AWS SDK

Dalam aplikasi, Anda dapat mengambil rahasia Anda dengan menelepon `GetSecretValue` atau `BatchGetSecretValue` di salah satu AWS SDK. Namun, kami menyarankan Anda menyimpan nilai rahasia Anda dengan menggunakan caching sisi klien. Rahasia caching meningkatkan kecepatan dan mengurangi biaya Anda.

Untuk aplikasi Rust, gunakan komponen Rust-based caching Secrets Manager atau panggil SDK langsung dengan GetSecretValue atau. BatchGetSecretValue

Contoh kode berikut menunjukkan cara mendapatkan nilai rahasia Secrets Manager.

Izin yang diperlukan: `secretsmanager:GetSecretValue`

```
async fn show_secret(client: &Client, name: &str) -> Result<(), Error> {
    let resp = client.get_secret_value().secret_id(name).send().await?;

    println!("Value: {}", resp.secret_string().unwrap_or("No value!"));

    Ok(())
}
```

Gunakan AWS Secrets Manager rahasia di Amazon Elastic Kubernetes Service

Untuk menampilkan secret from AWS Secrets Manager (ASCP) sebagai file yang dipasang di Amazon EKS Pods, Anda dapat menggunakan AWS Secrets and Configuration Provider untuk

Kubernetes Secrets Store CSI Driver. ASCP bekerja dengan Amazon Elastic Kubernetes Service 1.17+ yang menjalankan grup node Amazon EC2. AWS Fargate grup node tidak didukung. Dengan ASCP, Anda dapat menyimpan dan mengelola rahasia Anda di Secrets Manager dan kemudian mengambilnya melalui beban kerja Anda yang berjalan di Amazon EKS. Jika rahasia Anda berisi beberapa pasangan nilai kunci dalam format JSON, Anda dapat memilih mana yang akan dipasang di Amazon EKS. ASCP menggunakan JMESPath sintaks untuk menanyakan pasangan kunci-nilai dalam rahasia Anda. ASCP juga bekerja dengan parameter Parameter Store. ASCP menawarkan dua metode otentikasi dengan Amazon EKS Pendekatan pertama menggunakan Peran IAM untuk Akun Layanan (IRSA). Pendekatan kedua menggunakan Pod Identities. Setiap pendekatan memiliki manfaat dan kasus penggunaannya.

ASCP dengan Peran IAM untuk Akun Layanan (IRSA)

ASCP dengan Peran IAM untuk Akun Layanan (IRSA) memungkinkan Anda memasang rahasia dari file AWS Secrets Manager sebagai di Pod Amazon EKS Anda. Pendekatan ini cocok ketika:

- Anda perlu memasang rahasia sebagai file di Pod Anda.
- Anda menggunakan Amazon EKS versi 1.17 atau yang lebih baru dengan grup node Amazon EC2.
- Anda ingin mengambil pasangan kunci-nilai tertentu dari rahasia. JSON-formatted

Untuk informasi selengkapnya, lihat the section called “Integrasikan ASCP dengan IRSA untuk Amazon EKS”.

ASCP dengan Identitas Pod

ASCP dengan Identitas Pod EKS

Metode ASCP dengan Pod Identity meningkatkan keamanan dan menyederhanakan konfigurasi untuk mengakses rahasia di Amazon EKS. Pendekatan ini bermanfaat ketika:

- Anda memerlukan manajemen izin yang lebih terperinci di tingkat Pod.
- Anda menggunakan Amazon EKS versi 1.24 atau yang lebih baru.
- Anda ingin meningkatkan kinerja dan skalabilitas.

Untuk informasi selengkapnya, lihat the section called “Integrasikan ASCP dengan Pod Identity untuk Amazon EKS”.

Memilih pendekatan yang tepat

Pertimbangkan faktor-faktor berikut ketika memutuskan antara ASCP dengan IRSA dan ASCP dengan Pod Identity:

- Amazon EksVersion: Pod Identity memerlukan Amazon EKS 1.24+, sedangkan driver CSI bekerja dengan Amazon EKS 1.17+.
- Persyaratan keamanan: Pod Identity menawarkan kontrol yang lebih terperinci pada level Pod.
- Kinerja: Pod Identity umumnya berkinerja lebih baik di lingkungan skala tinggi.
- Kompleksitas: Pod Identity menyederhanakan pengaturan dengan menghilangkan kebutuhan akan akun layanan terpisah.

Pilih metode yang paling sesuai dengan kebutuhan spesifik Anda dan lingkungan Amazon EKS.

Instal ASCP untuk Amazon EKS

Bagian ini menjelaskan cara menginstal Penyedia AWS Rahasia dan Konfigurasi untuk Amazon EKS. Dengan ASCP, Anda dapat memasang rahasia dari Secrets Manager dan parameter dari AWS Systems Manager sebagai file di Amazon EKS Pods.

Prasyarat

- Klaster Amazon EKS
  - Versi 1.24 atau yang lebih baru untuk Pod Identity
  - Versi 1.17 atau yang lebih baru untuk IRSA
- Yang AWS CLI diinstal dan dikonfigurasi
- kubectl diinstal dan dikonfigurasi untuk klaster Amazon EKS Anda
- Helm (versi 3.0 atau yang lebih baru)

Instal dan konfigurasikan ASCP

ASCP tersedia di repositori GitHub secrets-store-csi-provider-aws. Repo juga berisi contoh file YAMAL untuk membuat dan memasang rahasia.

Selama instalasi, Anda dapat mengkonfigurasi ASCP untuk menggunakan titik akhir FIPS. Untuk daftar titik akhir, lihatthe section called “Titik akhir Manajer Rahasia”.

> **Add-on lingkup validasi skema**
>
> Selama instalasi, add-on hanya memvalidasi opsi konfigurasi AWS penyedia Anda terhadap skema. Itu tidak memvalidasi opsi Driver Secrets Store CSI. Untuk informasi selengkapnya tentang opsi konfigurasi Driver CSI Secrets Store, lihat Konfigurasi di bagan Helm README di situs web. GitHub

Untuk menginstal ASCP sebagai add-on EKS

1. Instal `eksctl` (petunjuk instalasi)
2. Jalankan perintah berikut untuk menginstal add-on dengan konfigurasi default:

   ```
   eksctl create addon --cluster <your_cluster> --name aws-secrets-store-csi-driver-provider
   ```

   Jika Anda ingin mengkonfigurasi add-on, jalankan perintah instalasi berikut sebagai gantinya:

   ```
   aws eks create-addon --cluster-name <your_cluster> --addon-name aws-secrets-store-csi-driver-provider --configuration-values 'file://path/to/config.yaml'
   ```

   File konfigurasi dapat berupa file YAMM atau JSON. Untuk melihat skema konfigurasi untuk add-on:

   a. Jalankan perintah berikut dan catat versi terbaru dari add-on:

      ```
      aws eks describe-addon-versions --addon-name aws-secrets-store-csi-driver-provider
      ```

   b. Jalankan perintah berikut untuk melihat skema konfigurasi add-on, ganti `<version>` dengan versi dari langkah sebelumnya:

      ```
      aws eks describe-addon-configuration --addon-name aws-secrets-store-csi-driver-provider --addon-version <version>
      ```

Untuk menginstal ASCP dengan menggunakan Helm

1. Untuk memastikan repo menunjuk ke grafik terbaru, gunakan `helm repo update.`

2. Instal bagan. Berikut ini adalah contoh `helm install` perintah:

```
helm install -n kube-system secrets-provider-aws aws-secrets-manager/secrets-store-csi-driver-provider-aws
```

   a. Untuk menggunakan endpoint FIPS, tambahkan tanda berikut: `--set useFipsEndpoint=true`

   b. Untuk mengonfigurasi pelambatan, tambahkan tanda berikut: `--set-json 'k8sThrottlingParams={"qps": "number of queries per second", "burst": "number of queries per second"}'`

   c. Jika Secrets Store CSI Driver sudah diinstal pada cluster Anda, tambahkan tanda berikut:`--set secrets-store-csi-driver.install=false`. Ini akan melewatkan menginstal Secrets Store CSI Driver sebagai dependensi.

Untuk menginstal dengan menggunakan YAMAL di repo

- Gunakan perintah berikut.

```
helm repo add secrets-store-csi-driver https://kubernetes-sigs.github.io/secrets-store-csi-driver/charts
helm install -n kube-system csi-secrets-store secrets-store-csi-driver/secrets-store-csi-driver
kubectl apply -f https://raw.githubusercontent.com/aws/secrets-store-csi-driver-provider-aws/main/deployment/aws-provider-installer.yaml
```

Verifikasi instalasi

Untuk memverifikasi instalasi kluster EKS Anda, driver Secrets Store CSI, dan plugin ASCP, ikuti langkah-langkah berikut:

1. Verifikasi kluster EKS:

```
eksctl get cluster --name clusterName
```

   Perintah ini harus mengembalikan informasi tentang cluster Anda.

2. Verifikasi instalasi driver Secrets Store CSI:

```
kubectl get pods -n kube-system -l app=secrets-store-csi-driver
```

Anda akan melihat Pod berjalan dengan nama seperti`csi-secrets-store-secrets-store-csi-driver-xxx`.

3. Verifikasi instalasi plugin ASCP:

   YAML installation

   ```
   $ kubectl get pods -n kube-system -l app=csi-secrets-store-provider-aws
   ```

   Contoh output:

   ```
   NAME                                     READY   STATUS    RESTARTS   AGE
   csi-secrets-store-provider-aws-12345      1/1     Running   0          2m
   ```

   Helm installation

   ```
   $  kubectl get pods -n kube-system -l app=secrets-store-csi-driver-provider-aws
   ```

   Contoh output:

   ```
   NAME                                                READY   STATUS    RESTARTS
    AGE
   secrets-provider-aws-secrets-store-csi-driver-provider-67890       1/1
    Running   0          2m
   ```

   Anda akan melihat Pod di `Running` negara bagian.

Setelah menjalankan perintah ini, jika semuanya diatur dengan benar, Anda akan melihat semua komponen berjalan tanpa kesalahan. Jika mengalami masalah apa pun, Anda mungkin perlu memecahkan masalah dengan memeriksa log Pod tertentu yang mengalami masalah.

Pemecahan masalah

1. Untuk memeriksa log penyedia ASCP, jalankan:

```
kubectl logs -n kube-system -l app=csi-secrets-store-provider-aws
```

2. Periksa status semua pod di `kube-system` namespace:

```
kubectl -n kube-system get pods
```

```
kubectl -n kube-system logs pod/PODID
```

Semua Pod yang terkait dengan driver CSI dan ASCP harus berada dalam status 'Berjalan'.

3. Periksa versi driver CSI:

```
kubectl get csidriver secrets-store.csi.k8s.io -o yaml
```

Perintah ini harus mengembalikan informasi tentang driver CSI yang diinstal.

Sumber daya tambahan

Untuk informasi selengkapnya tentang penggunaan ASCP dengan Amazon EKS, lihat sumber daya berikut:

- Menggunakan Identitas Pod dengan Amazon EKS
- AWS Secrets Store CSI Driver di GitHub

Gunakan CSI Penyedia AWS Rahasia dan Konfigurasi dengan Pod Identity untuk Amazon EKS

Integrasi AWS Secrets and Configuration Provider dengan Pod Identity Agent untuk Amazon Elastic Kubernetes Service memberikan peningkatan keamanan, konfigurasi yang disederhanakan, dan peningkatan kinerja untuk aplikasi yang berjalan di Amazon EKS. Pod Identity menyederhanakan autentikasi IAM untuk Amazon EKS saat mengambil rahasia dari Secrets Manager atau parameter dari Parameter Store. AWS Systems Manager

Amazon EKS Pod Identity merampingkan proses konfigurasi izin IAM untuk aplikasi Kubernetes dengan mengizinkan izin diatur secara langsung melalui antarmuka Amazon EKS, mengurangi jumlah langkah dan menghilangkan kebutuhan untuk beralih antara Amazon EKS dan layanan IAM. Pod Identity memungkinkan penggunaan peran IAM tunggal di beberapa cluster tanpa

memperbarui kebijakan kepercayaan dan mendukung tag sesi peran untuk kontrol akses yang lebih terperinci. Pendekatan ini tidak hanya menyederhanakan manajemen kebijakan dengan mengizinkan penggunaan kembali kebijakan izin di seluruh peran tetapi juga meningkatkan keamanan dengan mengaktifkan akses ke AWS sumber daya berdasarkan tag yang cocok.

Cara kerjanya

1. Pod Identity memberikan peran IAM ke Pod.
2. ASCP menggunakan peran ini untuk mengautentikasi dengan. Layanan AWS
3. Jika diotorisasi, ASCP mengambil rahasia yang diminta dan membuatnya tersedia untuk Pod.

Untuk informasi selengkapnya, lihat Memahami cara kerja Identitas Pod Amazon EKS di Panduan Pengguna Amazon EKS.

Prasyarat

> **Important**
>
> Pod Identity hanya didukung untuk Amazon EKS di cloud. Ini tidak didukung untuk Amazon EKS Anywhere, Layanan OpenShift Red Hat di AWS, atau cluster Kubernetes yang dikelola sendiri di instans Amazon EC2.

- Cluster Amazon EKS (versi 1.24 atau yang lebih baru)
- Akses ke AWS CLI dan Amazon EKS cluster melalui `kubectl`
- Akses ke dua Akun AWS (untuk akses lintas akun)

Instal Agen Identitas Pod Amazon EKS

Untuk menggunakan Pod Identity dengan klaster Anda, Anda harus menginstal add-on Amazon EKS Pod Identity Agent.

Untuk menginstal Agen Identitas Pod

- Instal add-on Pod Identity Agent di klaster Anda:

```
eksctl create addon \
  --name eks-pod-identity-agent \
```

```
  --cluster clusterName \
  --region region
```

Mengatur ASCP dengan Pod Identity

1. Buat kebijakan izin yang memberikan `secretsmanager:GetSecretValue` dan `secretsmanager:DescribeSecret` mengizinkan rahasia yang perlu diakses oleh Pod. Untuk contoh kebijakan, lihat the section called “Contoh: Izin untuk membaca dan menggambarkan rahasia individu”.
2. Buat peran IAM yang dapat diasumsikan oleh prinsipal layanan Amazon EKS untuk Pod Identity:

   JSON

   ```
   {
       "Version":"2012-10-17",
       "Statement": [
         {
           "Effect": "Allow",
           "Principal": {
             "Service": "pods.eks.amazonaws.com"
           },
           "Action": [
             "sts:AssumeRole",
             "sts:TagSession"
           ]
         }
       ]
     }
   ```

   Lampirkan kebijakan IAM ke peran:

   ```
   aws iam attach-role-policy \
     --role-name MY_ROLE \
     --policy-arn POLICY_ARN
   ```

3. Buat asosiasi Pod Identity. Sebagai contoh, lihat Membuat asosiasi Identitas Pod di Panduan Pengguna Amazon EKS
4. Buat `SecretProviderClass` yang menentukan rahasia mana yang akan dipasang di Pod:

```
kubectl apply -f https://raw.githubusercontent.com/aws/secrets-store-csi-driver-
provider-aws/main/examples/ExampleSecretProviderClass-PodIdentity.yaml
```

Perbedaan utama `SecretProviderClass` antara IRSA dan Pod Identity adalah parameter `usePodIdentity` opsional. Ini adalah bidang opsional yang menentukan pendekatan otentikasi. Ketika tidak ditentukan, default menggunakan Peran IAM untuk Akun Layanan (IRSA).

- Untuk menggunakan EKS Pod Identity, gunakan salah satu dari nilai-nilai ini:`"true", "True", "TRUE", "t", "T"`.
- Untuk secara eksplisit menggunakan IRSA, atur ke salah satu nilai ini:. `"false", "False", "FALSE", "f", or "F"`

5. Terapkan Pod yang memasang rahasia di bawah: `/mnt/secrets-store`

```
kubectl apply -f https://raw.githubusercontent.com/aws/secrets-store-csi-driver-
provider-aws/main/examples/ExampleDeployment-PodIdentity.yaml
```

6. Jika Anda menggunakan kluster Amazon EKS pribadi, pastikan VPC tempat cluster berada memiliki AWS STS titik akhir. Untuk informasi tentang membuat titik akhir, lihat Titik akhir VPC Antarmuka di AWS Identity and Access Management Panduan Pengguna.

Verifikasi pemasangan rahasia

Untuk memverifikasi bahwa rahasia dipasang dengan benar, jalankan perintah berikut:

```
kubectl exec -it $(kubectl get pods | awk '/pod-identity-deployment/{print $1}' | head
 -1) -- cat /mnt/secrets-store/MySecret
```

Untuk mengatur Amazon EKS Pod Identity untuk mengakses rahasia di Secrets Manager

1. Buat kebijakan izin yang memberikan `secretsmanager:GetSecretValue` dan `secretsmanager:DescribeSecret` mengizinkan rahasia yang perlu diakses oleh Pod. Untuk contoh kebijakan, lihat the section called “Contoh: Izin untuk membaca dan menggambarkan rahasia individu”.
2. Buat rahasia di Secrets Manager, jika Anda belum memilikinya.

Pemecahan Masalah

Anda dapat melihat sebagian besar kesalahan dengan menjelaskan penerapan Pod.

Untuk melihat pesan kesalahan untuk penampung Anda

1. Dapatkan daftar nama Pod dengan perintah berikut. Jika Anda tidak menggunakan namespace default, gunakan. `-n` *`NAMESPACE`*

   ```
   kubectl get pods
   ```

2. Untuk mendeskripsikan Pod, dalam perintah berikut, *`PODID`* gunakan ID Pod dari Pod yang Anda temukan di langkah sebelumnya. Jika Anda tidak menggunakan namespace default, gunakan. `-n` *`NAMESPACE`*

   ```
   kubectl describe pod/PODID
   ```

Untuk melihat kesalahan untuk ASCP

- Untuk menemukan informasi selengkapnya di log provider, pada perintah berikut, *`PODID`* gunakan ID Pod csi-secrets-store-provider-aws.

  ```
  kubectl -n kube-system get pods
  kubectl -n kube-system logs pod/PODID
  ```

Gunakan AWS Rahasia dan Penyedia Konfigurasi CSI dengan Peran IAM untuk Akun Layanan (IRSA)

Topik

- Prasyarat
- Mengatur kontrol akses
- Identifikasi rahasia mana yang akan dipasang
- Pemecahan Masalah

Prasyarat

- Cluster Amazon EKS (versi 1.17 atau yang lebih baru)
- Akses ke AWS CLI dan Amazon EKS cluster melalui `kubectl`

Mengatur kontrol akses

ASCP mengambil Amazon EKS Pod Identity dan menukarnya dengan peran IAM. Anda menetapkan izin dalam kebijakan IAM untuk peran IAM tersebut. Ketika ASCP mengasumsikan peran IAM, ia mendapat akses ke rahasia yang Anda otorisasi. Kontainer lain tidak dapat mengakses rahasia kecuali Anda juga mengaitkannya dengan peran IAM.

Untuk memberikan Amazon EKS Pod akses ke rahasia di Secrets Manager

1. Buat kebijakan izin yang memberikan `secretsmanager:GetSecretValue` dan `secretsmanager:DescribeSecret` mengizinkan rahasia yang perlu diakses oleh Pod. Untuk contoh kebijakan, lihat the section called “Contoh: Izin untuk membaca dan menggambarkan rahasia individu”.
2. Buat penyedia OpenID Connect (OIDC) IAM untuk cluster jika Anda belum memilikinya. Untuk informasi selengkapnya, lihat Membuat penyedia IAM OIDC untuk klaster Anda di Panduan Pengguna Amazon EKS.
3. Buat peran IAM untuk akun layanan dan lampirkan kebijakan ke dalamnya. Untuk informasi selengkapnya, lihat Membuat peran IAM untuk akun layanan di Panduan Pengguna Amazon EKS.
4. Jika Anda menggunakan kluster Amazon EKS pribadi, pastikan VPC tempat cluster berada memiliki AWS STS titik akhir. Untuk informasi tentang membuat titik akhir, lihat Titik akhir VPC Antarmuka di AWS Identity and Access Management Panduan Pengguna.

Identifikasi rahasia mana yang akan dipasang

Untuk menentukan rahasia mana yang dipasang ASCP di Amazon EKS sebagai file di sistem file, Anda membuat file YAMAL. the section called “SecretProviderClass” `SecretProviderClass`Daftar rahasia untuk dipasang dan nama file untuk dipasang sebagai. `SecretProviderClass`Harus berada di namespace yang sama dengan Amazon EKS Pod yang direferensikannya.

Pasang rahasia sebagai file

Instruksi berikut menunjukkan cara me-mount rahasia sebagai file menggunakan contoh file YAMAL ExampleSecretProviderClass.yamldan ExampleDeployment.yamlfile.

Untuk memasang rahasia di Amazon EKS

1. Terapkan `SecretProviderClass` ke Pod:

```
kubectl apply -f ExampleSecretProviderClass.yaml
```

2. Menerapkan Pod Anda:

```
kubectl apply -f ExampleDeployment.yaml
```

3. ASCP memasang file.

Pemecahan Masalah

Anda dapat melihat sebagian besar kesalahan dengan menjelaskan penerapan Pod.

Untuk melihat pesan galat untuk penampung

1. Dapatkan daftar nama Pod dengan perintah berikut. Jika Anda tidak menggunakan namespace default, gunakan. `-n` *`nameSpace`*

```
kubectl get pods
```

2. Untuk mendeskripsikan Pod, dalam perintah berikut, *`podId`* gunakan ID Pod dari Pod yang Anda temukan di langkah sebelumnya. Jika Anda tidak menggunakan namespace default, gunakan. `-n` *`nameSpace`*

```
kubectl describe pod/podId
```

Untuk melihat kesalahan untuk ASCP

- Untuk menemukan informasi selengkapnya di log provider, pada perintah berikut, *`podId`* gunakan ID Pod csi-secrets-store-provider-aws.

```
kubectl -n kube-system get pods
```

```
kubectl -n kube-system logs Pod/podId
```

- Verifikasi bahwa **`SecretProviderClass`** CRD diinstal:

  ```
  kubectl get crd secretproviderclasses.secrets-store.csi.x-k8s.io
  ```

  Perintah ini harus mengembalikan informasi tentang definisi sumber daya `SecretProviderClass` khusus.

- Verifikasi bahwa SecretProviderClass objek telah dibuat.

  ```
  kubectl get secretproviderclass SecretProviderClassName -o yaml
  ```

AWS Contoh kode Penyedia Rahasia dan Konfigurasi

Otentikasi ASCP dan contoh kontrol akses

Contoh: Kebijakan IAM yang mengizinkan layanan Amazon EKS Pod Identity (pods.eks.amazonaws.com) untuk mengambil peran dan menandai sesi:

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "pods.eks.amazonaws.com"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:TagSession"
      ]
    }
  ]
}
```

SecretProviderClass

Anda menggunakan YAML untuk menjelaskan rahasia mana yang akan dipasang di Amazon EKS menggunakan ASCP. Sebagai contoh, lihat the section called “SecretProviderClass penggunaan”.

SecretProviderClass Struktur YAML

```
apiVersion: secrets-store.csi.x-k8s.io/v1
kind: SecretProviderClass
metadata:
   name: name
spec:
  provider: aws
  parameters:
    region:
    failoverRegion:
    pathTranslation:
    usePodIdentity:
    preferredAddressType:
    objects:
```

Bidang parameter berisi rincian permintaan pemasangan:

region

(Opsional) Wilayah AWS Rahasianya. Jika Anda tidak menggunakan bidang ini, ASCP mencari Region dari anotasi pada node. Pencarian ini menambahkan overhead untuk permintaan mount, jadi sebaiknya Anda menyediakan Wilayah untuk cluster yang menggunakan sejumlah besar Pod.

Jika Anda juga menentukan`failoverRegion`, ASCP mencoba mengambil rahasia dari kedua Wilayah. Jika salah satu Wilayah mengembalikan kesalahan 4xx, misalnya untuk masalah otentikasi, ASCP tidak memasang salah satu rahasia. Jika rahasia berhasil diambil dari`region`, maka ASCP memasang nilai rahasia itu. Jika rahasia tidak berhasil diambil dari`region`, tetapi berhasil diambil dari`failoverRegion`, maka ASCP memasang nilai rahasia itu.

Wilayah Failover

(Opsional) Jika Anda menyertakan bidang ini, ASCP mencoba mengambil rahasia dari Wilayah yang ditentukan dalam `region` dan bidang ini. Jika salah satu Wilayah mengembalikan kesalahan 4xx, misalnya untuk masalah otentikasi, ASCP tidak memasang salah satu rahasia. Jika rahasia berhasil diambil dari`region`, maka ASCP memasang nilai rahasia itu. Jika rahasia tidak berhasil diambil dari`region`, tetapi berhasil diambil dari`failoverRegion`, maka ASCP

memasang nilai rahasia itu. Untuk contoh cara menggunakan bidang ini, lihatMulti-Region failover rahasia.

Terjemahan jalur

(Opsional) Karakter substitusi tunggal untuk digunakan jika nama file di Amazon EKS akan berisi karakter pemisah jalur, seperti garis miring (/) di Linux. ASCP tidak dapat membuat file terpasang yang berisi karakter pemisah jalur. Sebaliknya, ASCP menggantikan karakter pemisah jalur dengan karakter yang berbeda. Jika Anda tidak menggunakan bidang ini, karakter pengganti adalah garis bawah (_), jadi misalnya, `My/Path/Secret` mount as`My_Path_Secret`.

Untuk mencegah substitusi karakter, masukkan string`False`.

menggunakan PodIdentity

(Opsional) Menentukan pendekatan otentikasi. Jika tidak ditentukan, default adalah Peran IAM untuk Akun Layanan (IRSA) (IRSA).

- Untuk menggunakan EKS Pod Identity, gunakan salah satu nilai berikut: `"true"` “,`"True"`,`"TRUE"`,`"t"`, atau`"T"`.
- Untuk menggunakan IRSA secara eksplisit, atur ke salah satu nilai berikut:`"false"`,`"False"`,`"FALSE"`,`"f"`, atau `"F"` “=.

lebih disukai AddressType

(Opsional) Menentukan jenis alamat IP pilihan untuk komunikasi titik akhir Agen Identity Pod. Bidang ini hanya berlaku saat menggunakan fitur EKS Pod Identity dan akan diabaikan saat menggunakan Peran IAM untuk Layanan tidak peka huruf besar/ Accounts.Values kecil. Nilai yang valid adalah:

- `"ipv4"`, `"IPv4"` “, atau `"IPV4"` — Paksa penggunaan titik akhir IPv4 Agen Identity Pod
- `"ipv6"`,`"IPv6"`, atau `"IPV6"` — Paksa penggunaan titik akhir IPv6 Pod Identity Agent
- tidak ditentukan — Gunakan pemilihan titik akhir otomatis, coba titik akhir IPv4 terlebih dahulu dan kembali ke titik akhir IPv6 jika IPv4 gagal

objek

Sebuah string yang berisi deklarasi YAML dari rahasia yang akan dipasang. Sebaiknya gunakan string multi-baris YAML atau karakter pipe (|).

Nama Objek

Wajib. Menentukan nama rahasia atau parameter yang akan diambil. Untuk Manajer Rahasia ini adalah `SecretId` parameter dan dapat berupa nama ramah atau ARN lengkap rahasia.

Untuk Penyimpanan Parameter SSM, ini adalah parameter dan dapat berupa nama atau ARN lengkap parameter. `Name`

Tipe Objek

Diperlukan jika Anda tidak menggunakan Manajer Rahasia ARN untuk`objectName`. Bisa salah satu `secretsmanager` atau`ssmparameter`.

ObjekTalias

(Opsional) Nama file rahasia di Amazon EKS Pod. Jika Anda tidak menentukan bidang ini, akan `objectName` muncul sebagai nama file.

Izin berkas

(Opsional) String oktal 4 digit yang menentukan izin file untuk memasang rahasia. Jika Anda tidak menentukan bidang ini, itu akan menjadi default`"0644"`.

Versi Objek

(Opsional) ID versi rahasia. Tidak disarankan karena Anda harus memperbarui ID versi setiap kali Anda memperbarui rahasia. Secara default versi terbaru digunakan. Jika Anda menyertakan `afailoverRegion`, bidang ini mewakili yang utama`objectVersion`.

objek VersionLabel

(Opsional) Alias untuk versi. Defaultnya adalah versi terbaru AWSCURRENT. Untuk informasi selengkapnya, lihat the section called “Versi rahasia”. Jika Anda menyertakan `afailoverRegion`, bidang ini mewakili yang utama`objectVersionLabel`.

JMEpath

(Opsional) Peta kunci dalam rahasia file yang akan dipasang di Amazon EKS. Untuk menggunakan bidang ini, nilai rahasia Anda harus dalam format JSON. Jika Anda menggunakan bidang ini, Anda harus menyertakan subbidang `path` dan`objectAlias`.

path

Kunci dari pasangan kunci-nilai dalam JSON dari nilai rahasia. Jika bidang berisi tanda hubung, gunakan tanda kutip tunggal untuk menghindarinya, misalnya: `path: '"hyphenated-path"'`

ObjekTalias

Nama file yang akan dipasang di Amazon EKS Pod. Jika bidang berisi tanda hubung, gunakan tanda kutip tunggal untuk menghindarinya, misalnya: `objectAlias: '"hyphenated-alias"'`

Izin berkas

(Opsional) String oktal 4 digit yang menentukan izin file untuk memasang rahasia. Jika Anda tidak menentukan bidang ini, itu akan secara default ke izin file objek induk.

Objek Failover

(Opsional) Jika Anda menentukan bidang ini, ASCP mencoba mengambil rahasia yang ditentukan di primer `objectName` dan rahasia yang ditentukan di `failoverObject` `objectName` sub-bidang. Jika salah satu mengembalikan kesalahan 4xx, misalnya untuk masalah otentikasi, ASCP tidak memasang salah satu rahasia. Jika rahasia berhasil diambil dari primer`objectName`, maka ASCP memasang nilai rahasia itu. Jika rahasia tidak berhasil diambil dari primer`objectName`, tetapi berhasil diambil dari failover`objectName`, maka ASCP memasang nilai rahasia itu. Jika Anda menyertakan bidang ini, Anda harus menyertakan bidang tersebut`objectAlias`. Untuk contoh cara menggunakan bidang ini, lihatFailover ke rahasia yang berbeda.

Anda biasanya menggunakan bidang ini ketika rahasia failover bukan replika. Untuk contoh cara menentukan replika, lihatMulti-Region failover rahasia.

Nama Objek

Nama atau ARN lengkap dari rahasia failover. Jika Anda menggunakan ARN, Wilayah di ARN harus cocok dengan bidang tersebut`failoverRegion`.

Versi Objek

(Opsional) ID versi rahasia. Harus cocok dengan yang utama`objectVersion`. Tidak disarankan karena Anda harus memperbarui ID versi setiap kali Anda memperbarui rahasia. Secara default versi terbaru digunakan.

objek VersionLabel

(Opsional) Alias untuk versi. Defaultnya adalah versi terbaru AWSCURRENT. Untuk informasi selengkapnya, lihat the section called “Versi rahasia”.

Buat SecretProviderClass konfigurasi dasar untuk memasang rahasia di Amazon EKS Pod Anda.

Pod Identity

SecretProviderClass untuk menggunakan rahasia di cluster Amazon EKS yang sama:

```
apiVersion: secrets-store.csi.x-k8s.io/v1
```

```
kind: SecretProviderClass
metadata:
  name: aws-secrets-manager
spec:
  provider: aws
  parameters:
    objects: |
      - objectName: "mySecret"
        objectType: "secretsmanager"
    usePodIdentity: "true"
```

IRSA

```
apiVersion: secrets-store.csi.x-k8s.io/v1
kind: SecretProviderClass
metadata:
  name: deployment-aws-secrets
spec:
  provider: aws
  parameters:
    objects: |
        - objectName: "MySecret"
          objectType: "secretsmanager"
```

SecretProviderClass penggunaan

Gunakan contoh ini untuk membuat SecretProviderClass konfigurasi untuk skenario yang berbeda.

Contoh: Pasang rahasia dengan nama atau ARN

Contoh ini menunjukkan cara memasang tiga jenis rahasia yang berbeda:

- Rahasia yang ditentukan oleh ARN penuh
- Rahasia yang ditentukan dengan nama
- Versi spesifik dari sebuah rahasia

```
apiVersion: secrets-store.csi.x-k8s.io/v1
kind: SecretProviderClass
metadata:
  name: aws-secrets
```

```
spec:
  provider: aws
  parameters:
    objects: |
      - objectName: "arn:aws:secretsmanager:us-east-2:777788889999:secret:MySecret2-
d4e5f6"
      - objectName: "MySecret3"
        objectType: "secretsmanager"
      - objectName: "MySecret4"
        objectType: "secretsmanager"
        objectVersionLabel: "AWSCURRENT"
```

Contoh: Pasang pasangan kunci-nilai dari rahasia

Contoh ini menunjukkan cara memasang pasangan kunci-nilai tertentu dari rahasia: JSON-formatted

```
apiVersion: secrets-store.csi.x-k8s.io/v1
kind: SecretProviderClass
metadata:
  name: aws-secrets
spec:
  provider: aws
  parameters:
    objects: |
      - objectName: "arn:aws:secretsmanager:us-east-2:777788889999:secret:MySecret-
a1b2c3"
        jmesPath:
            - path: username
              objectAlias: dbusername
            - path: password
              objectAlias: dbpassword
```

Contoh: Pasang rahasia dengan izin file

Contoh ini menunjukkan cara memasang rahasia dengan izin file tertentu

```
apiVersion: secrets-store.csi.x-k8s.io/v1
kind: SecretProviderClass
metadata:
  name: aws-secrets
spec:
  provider: aws
  parameters:
```

```
    objects: |
      - objectName: "mySecret"
        objectType: "secretsmanager"
        filePermission: "0600"
        jmesPath:
            - path: username
              objectAlias: dbusername
              filePermission: "0400"
```

Contoh: Contoh konfigurasi failover

Contoh-contoh ini menunjukkan cara mengkonfigurasi failover untuk rahasia.

Multi-Region failover rahasia

Contoh ini menunjukkan cara mengonfigurasi failover otomatis untuk rahasia yang direplikasi di beberapa Wilayah:

```
apiVersion: secrets-store.csi.x-k8s.io/v1
kind: SecretProviderClass
metadata:
  name: aws-secrets
spec:
  provider: aws
  parameters:
    region: us-east-1
    failoverRegion: us-east-2
    objects: |
      - objectName: "MySecret"
```

Failover ke rahasia yang berbeda

Contoh ini menunjukkan cara mengkonfigurasi failover ke rahasia yang berbeda (bukan replika):

```
apiVersion: secrets-store.csi.x-k8s.io/v1
kind: SecretProviderClass
metadata:
  name: aws-secrets
spec:
  provider: aws
  parameters:
    region: us-east-1
```

```
    failoverRegion: us-east-2
    objects: |
      - objectName: "arn:aws:secretsmanager:us-east-1:777788889999:secret:MySecret-
a1b2c3"
        objectAlias: "MyMountedSecret"
        failoverObject:
          - objectName: "arn:aws:secretsmanager:us-
east-2:777788889999:secret:MyFailoverSecret-d4e5f6"
```

Sumber daya tambahan

Untuk informasi selengkapnya tentang menggunakan ASCP dengan Amazon EKS, lihat sumber daya berikut:

- Menggunakan Identitas Pod dengan Amazon EKS
- Menggunakan Penye AWS dia Rahasia dan Konfigurasi
- AWS Secrets Store CSI Driver aktif GitHub

Gunakan AWS Secrets Manager rahasia dalam AWS Lambda fungsi

AWS Lambda adalah layanan komputasi tanpa server yang memungkinkan Anda menjalankan kode tanpa menyediakan atau mengelola server. Parameter Store, kemampuan AWS Systems Manager, menyediakan penyimpanan hierarkis yang aman untuk manajemen data konfigurasi dan manajemen rahasia. Anda dapat menggunakan AWS Parameter dan Rahasia Lambda Extension untuk mengambil dan menyimpan AWS Secrets Manager rahasia dan parameter Parameter Store dalam fungsi Lambda tanpa menggunakan SDK. Untuk informasi mendetail tentang penggunaan ekstensi ini, lihat Menggunakan rahasia Secrets Manager di fungsi Lambda di Panduan Pengembang Lambda.

Menggunakan rahasia Secrets Manager dengan Lambda

Panduan Pengembang Lambda memberikan instruksi komprehensif untuk menggunakan rahasia Secrets Manager dalam fungsi Lambda. Untuk memulai:

1. Ikuti tutorial langkah demi langkah di Gunakan rahasia Secrets Manager dalam fungsi Lambda, yang meliputi:
   - Membuat fungsi Lambda dengan runtime pilihan Anda (Python,, Java) Node.js

- Menambahkan AWS Parameter dan Rahasia Ekstensi Lambda sebagai lapisan
- Mengkonfigurasi izin yang diperlukan
- Menulis kode untuk mengambil rahasia dari ekstensi
- Menguji fungsi Anda

2. Pelajari tentang variabel lingkungan untuk mengonfigurasi perilaku ekstensi, termasuk pengaturan cache dan batas waktu
3. Memahami praktik terbaik untuk bekerja dengan rotasi rahasia

Menggunakan Secrets Manager dan Lambda di VPC

Jika fungsi Lambda Anda berjalan di VPC, Anda perlu membuat titik akhir VPC sehingga ekstensi dapat melakukan panggilan ke Secrets Manager. Untuk informasi selengkapnya, lihat the section called “Titik akhir VPC (AWS PrivateLink)”.

Menggunakan AWS Parameter dan Rahasia Ekstensi Lambda

Ekstensi dapat mengambil rahasia Secrets Manager dan parameter Parameter Store. Untuk informasi rinci tentang penggunaan parameter Parameter Store dengan ekstensi, lihat Menggunakan parameter Parameter Menyimpan dalam fungsi Lambda di AWS Systems Manager Panduan Pengguna.

Dokumentasi Systems Manager meliputi:

- Penjelasan terperinci tentang cara kerja ekstensi dengan Parameter Store
- Petunjuk untuk menambahkan ekstensi ke fungsi Lambda
- Variabel lingkungan untuk mengkonfigurasi ekstensi
- Contoh perintah untuk mengambil parameter
- Daftar lengkap ARN ekstensi untuk semua arsitektur dan wilayah yang didukung

Menggunakan AWS Penyedia Kredensia Beban Kerja

Bagaimana AWS Penyedia KredenSIAL Beban Kerja berfungsi

Penyedia AWS Kredensi Beban Kerja (sebelumnya AWS Secrets Manager Agen) menyediakan layanan HTTP sisi klien yang membantu Anda menstandarkan cara Anda menggunakan rahasia

dari Secrets Manager di seluruh lingkungan komputasi Anda. Anda dapat menggunakannya dengan layanan berikut:

- AWS Lambda
- Amazon Elastic Container Service
- Layanan Amazon Elastic Kubernetes
- Amazon Elastic Compute Cloud

Penye AWS dia KredenSIAL Beban Kerja mengambil dan menyimpan rahasia dalam memori, memungkinkan aplikasi Anda mendapatkan rahasia dari localhost alih-alih melakukan panggilan langsung ke Manajer Rahasia. Penyedia AWS Kredensi Beban Kerja hanya dapat membaca rahasia —tidak dapat memodifikasinya.

Penyedia AWS Kredensi Beban Kerja adalah sumber terbuka. Kode sumber, petunjuk instalasi, dan informasi rilis terbaru tersedia di GitHub.

> **Important**
>
> Penyedia AWS KredenSIAL Beban Kerja menggunakan AWS kredenSIAL dari lingkungan Anda untuk memanggil Manajer Rahasia. Ini termasuk perlindungan terhadap Pemalsuan Permintaan Sisi Server (SSRF) untuk membantu meningkatkan keamanan rahasia. Penyedia AWS Kredensi Beban Kerja menggunakan pertukaran ML-KEM kunci pasca-kuantum sebagai pertukaran kunci prioritas tertinggi secara default.

Memahami AWS Caching Penyedia KredenSIAL Beban Kerja

Penyedia AWS Kredensi Beban Kerja menggunakan cache dalam memori yang disetel ulang saat Penyedia Kredensi Beban AWS Kerja dimulai ulang. Ini secara berkala menyegarkan nilai-nilai rahasia yang di-cache berdasarkan hal berikut:

- Frekuensi refresh default (TTL) adalah 300 detik
- Anda dapat memodifikasi TTL menggunakan file konfigurasi
- Refresh terjadi saat Anda meminta rahasia setelah TTL kedaluwarsa

> **Note**
>
> Penyedia AWS Kredensi Beban Kerja tidak menyertakan pembatalan cache. Jika rahasia berputar sebelum entri cache berakhir, Penyedia Kre AWS densi Beban Kerja mungkin mengembalikan nilai rahasia basi.

Penye AWS dia KredenSIAL Beban Kerja mengembalikan nilai rahasia dalam format yang sama dengan respons`GetSecretValue`. Nilai rahasia tidak dienkripsi dalam cache.

Topik

- Unduh AWS Penyedia Kredensia Beban Kerja
- Bangun AWS Penyedia KredenSIAL Beban Kerja dari sumber
- Instal AWS Penyedia Kredensia Beban Kerja
- Mengambil rahasia dengan AWS Penyedia Kredensia Beban Kerja
- Memahami parameter RefreshNow
- Mengambil rahasia di seluruh akun dengan rantai peran
- Pre-fetch rahasia saat startup
- Konfigurasikan AWS Penyedia Kredensia Beban Kerja
- Fitur opsional
- Pencatatan log
- Pertimbangan keamanan

Unduh AWS Penyedia Kredensia Beban Kerja

Untuk mengunduh biner bawaan dari Penyedia Kreden AWS sifikasi Beban Kerja, gunakan tautan berikut. Rilis untuk Windows ditandatangani.

Platform	Arsitektur	Unduh URL	Checksum () SHA-256
Linux	x86-64	Unduh untuk Linux x86-64 (3.1.1)	471c1978f8bf63a0aefb425e974ac3c816c7e04feb302236512eea375160de4

Platform	Arsitektur	Unduh URL	Checksum () SHA-256
Linux	AArch64	Unduh untuk Linux AArch64 (3.1.1)	8b09dc4e58b84379f580a9ae179940bcb3de0338421f638a43376bf73380ffb6
Windows	x86-64	Unduh untuk Windows x86-64 (3.1.1)	5fbdac4017e630556b94b90e5ed9ed11be69ac7a47e67655e20181ce990dbe12

Bangun AWS Penyedia KredenSIAL Beban Kerja dari sumber

Atau, Anda dapat membangun Penyedia Kreden AWS SIAL Beban Kerja dari sumber. Sebelum Anda mulai, pastikan Anda memiliki alat dan Rust alat pengembangan standar yang diinstal untuk platform Anda.

> **Note**
>
> Membangun penyedia dengan `fips` fitur yang diaktifkan di macOS saat ini memerlukan solusi berikut:
>
> - Buat variabel lingkungan `SDKROOT` yang disebut yang disetel ke hasil berjalan `xcrun --show-sdk-path`

RPM-based systems

Untuk membangun RPM-based sistem

1. Gunakan `install` skrip yang disediakan di repositori.

   Skrip menghasilkan token SSRF acak saat startup dan menyimpannya dalam file`/var/run/awssmatoken`. Token dapat dibaca oleh `aws-wcp-token` grup yang dibuat skrip instalasi.
2. Untuk mengizinkan aplikasi Anda membaca file token, Anda perlu menambahkan akun pengguna yang dijalankan aplikasi Anda ke `aws-wcp-token` grup. Misalnya, Anda dapat

memberikan izin bagi aplikasi Anda untuk membaca file token dengan perintah usermod berikut, di *<APP_USER>* mana ID pengguna tempat aplikasi Anda berjalan.

```
sudo usermod -aG aws-wcp-token <APP_USER>
```

Instal alat pengembangan

Pada RPM-based sistem seperti AL2023, instal grup Alat Pengembangan:

```
sudo yum -y groupinstall "Development Tools"
```

3. Instal Rust

   Ikuti petunjuk di Instal Rust di dokumentasi Rust:

   ```
   curl --proto '=https' --tlsv1.2 -sSf https://sh.rustup.rs | sh # Follow the on-screen instructions
   . "$HOME/.cargo/env"
   ```

4. Bangun penyedia

   Bangun Penye AWS dia KredenSIAL Beban Kerja menggunakan perintah pembuatan kargo:

   ```
   cargo build --release
   ```

   Anda akan menemukan yang dapat dieksekusi di bawah`target/release/aws-workload-credentials-provider`.

Debian-based systems

Untuk membangun Debian-based sistem

1. Instal alat pengembangan

   Pada Debian-based sistem seperti Ubuntu, instal paket build-essential:

   ```
   sudo apt install build-essential
   ```

2. Instal Rust

Ikuti petunjuk di Instal Rust di dokumentasi Rust:

```
curl --proto '=https' --tlsv1.2 -sSf https://sh.rustup.rs | sh # Follow the on-
screen instructions
. "$HOME/.cargo/env"
```

3. Bangun penyedia

   Bangun Penye AWS dia KredenSIAL Beban Kerja menggunakan perintah pembuatan kargo:

   ```
   cargo build --release
   ```

   Anda akan menemukan yang dapat dieksekusi di bawah`target/release/aws-workload-credentials-provider`.

Windows

Untuk membangun di Windows

1. Menyiapkan lingkungan pengembangan

   Ikuti petunjuk di Meng atur lingkungan pengembang Anda di Windows untuk Rust di dokumentasi Microsoft Windows.

2. Bangun penyedia

   Bangun Penye AWS dia KredenSIAL Beban Kerja menggunakan perintah pembuatan kargo:

   ```
   cargo build --release
   ```

   Anda akan menemukan yang dapat dieksekusi di bawah`target/release/aws-workload-credentials-provider.exe`.

Cross-compile natively

Untuk mengkompilasi silang secara asli

1. Instal alat kompilasi silang

Instal `cargo-xwin`:

```
cargo install cargo-xwin
```

2. Tambahkan target build Rust

   Instal target build Windows MSVC:

   ```
   rustup target add x86_64-pc-windows-msvc
   ```

3. Bangun untuk Windows

   Cross-compile penyedia untuk Windows:

   ```
   cargo xwin build --release --target x86_64-pc-windows-msvc
   ```

   Anda akan menemukan yang dapat dieksekusi di`target/x86_64-pc-windows-msvc/release/aws-workload-credentials-provider.exe`.

Instal AWS Penyedia Kredensia Beban Kerja

Pilih lingkungan komputasi Anda dari opsi instalasi berikut.

Amazon EC2

Pada Linux instans Amazon EC2, Anda dapat menginstal Penyedia Kreden AWS sifikasi Beban Kerja menggunakan paket RPM (tersedia untuk AL2023) atau skrip penginstalan manual.

Opsi A: Instal paket RPM (AL2023)

1. Instal paket

   Instal paket RPM. Manajer paket menemukan versi terbaru untuk Anda:

   ```
   sudo dnf install aws-workload-credentials-provider
   ```

   Paket RPM secara otomatis mengatur hal berikut:

- Pengguna `aws-wcp` layanan, bersama dengan `aws-wcp-token` kelompok `awscreds` dan.
- Skrip biner dan pembantu di`/opt/aws/workload-credentials-provider/`.
- Layanan AWS Secrets Manager dan tokensystemd, yang dimulai secara otomatis.
- Token SSRF acak di`/var/run/awssmatoken`.

Kemampuan Secrets Manager tersedia segera setelah instalasi. Saat Anda menginstal dengan paket RPM, Penyedia Kre AWS densi Beban Kerja membaca opsi konfigurasi dari jalur default`/etc/aws-workload-credentials-provider/config.toml`. Untuk menyesuaikan konfigurasi, buat atau edit file itu.

2. (Opsional) Aktifkan kemampuan Manajemen Sertifikat

   Untuk mengambil dan menyegarkan sertifikat dari AWS Certificate Manager, lihat Otomatisasi sertifikat di Panduan AWS Certificate Manager Pengguna.

3. Konfigurasikan izin aplikasi

   Untuk mengizinkan aplikasi Anda membaca file token SSRF, tambahkan akun pengguna aplikasi ke `aws-wcp-token` grup:

   ```
   sudo usermod -aG aws-wcp-token APP_USER
   ```

   Ganti *APP_USER* dengan ID pengguna tempat aplikasi Anda berjalan.

4. (Opsional) Berikan akses ke log penyedia

   Untuk membaca log penyedia, tambahkan pengguna Anda ke `awscreds` grup, lalu keluar dan masuk kembali (atau jalankan`newgrp awscreds`) agar perubahan diterapkan:

   ```
   sudo usermod -aG awscreds APP_USER
   ```

> **Copot pemasangan AWS Penyedia Kredensia Beban Kerja**
>
> Untuk menghapus Penyedia Kre AWS denSIAL Beban Kerja, jalankan`sudo dnf remove aws-workload-credentials-provider`. Menghapus instalasi

menghentikan semua layanan dan menghapus binari dan unit layanan. Proses uninstall mempertahankan `aws-wcp` pengguna, grup, dan direktori log.

Opsi B: Jalankan skrip instal (apa saja Linux)

1. Arahkan ke direktori konfigurasi

   Ubah ke direktori konfigurasi:

   ```
   cd aws_workload_credentials_provider_common/configuration
   ```

2. Jalankan skrip instalasi

   Jalankan `install` skrip yang disediakan di repositori.

   Skrip menghasilkan token SSRF acak saat startup dan menyimpannya dalam file`/var/run/awssmatoken`. Token dapat dibaca oleh `aws-wcp-token` grup yang dibuat skrip instalasi.

3. Konfigurasikan izin aplikasi

   Tambahkan akun pengguna yang dijalankan aplikasi Anda ke `aws-wcp-token` grup:

   ```
   sudo usermod -aG aws-wcp-token APP_USER
   ```

   Ganti *APP_USER* dengan ID pengguna tempat aplikasi Anda berjalan.

Container Sidecar

Anda dapat menjalankan Penyedia Kreden AWS sifikasi Beban Kerja sebagai wadah sidecar di samping aplikasi Anda dengan menggunakan Docker. Kemudian aplikasi Anda dapat mengambil rahasia dari server HTTP lokal yang disediakan oleh Penyedia Kreden AWS SIAL Beban Kerja. Untuk informasi tentang Docker, lihat dokumentasi Docker.

Untuk membuat wadah sidecar untuk AWS Penyedia Kredensia Beban Kerja

1. Buat penyedia Dockerfile

   Buat Dockerfile untuk wadah sidecar Penyedia Kreden AWS sia Beban Kerja:

   ```
   # Use the latest Debian image as the base
   ```

```
FROM debian:latest

# Set the working directory inside the container
WORKDIR /app

# Copy the Workload Credentials Provider binary to the container
COPY aws-workload-credentials-provider .

# Install any necessary dependencies
RUN apt-get update && apt-get install -y ca-certificates

# Set the entry point to run the provider
ENTRYPOINT ["./aws-workload-credentials-provider", "sm", "start"]
```

2. Buat aplikasi Dockerfile

   Buat Dockerfile untuk aplikasi klien Anda.

3. Buat file Docker Compose

   Buat file Docker Compose untuk menjalankan kedua wadah dengan antarmuka jaringan bersama:

> **Important**
>
> Anda harus memuat AWS kredenSIAL dan token SSRF agar aplikasi dapat menggunakan Penyedia KredenSIAL Beb AWS an Kerja. Untuk Amazon EKS dan Amazon ECS, lihat berikut ini:
>
> - Kelola akses di Panduan Pengguna Amazon EKS
> - Peran IAM tugas Amazon ECS dalam Panduan Pengembang Amazon ECS

```
version: '3'
services:
    client-application:
    container_name: client-application
    build:
        context: .
        dockerfile: Dockerfile.client
    command: tail -f /dev/null  # Keep the container running
```

```
    workload-credentials-provider:
    container_name: workload-credentials-provider
    build:
        context: .
        dockerfile: Dockerfile.provider
    network_mode: "container:client-application"  # Attach to the client-
application container's network
    depends_on:
        - client-application
```

4. Salin penyedia biner

   Salin `aws-workload-credentials-provider` biner ke direktori yang sama yang berisi file Dockerfiles dan Docker Compose Anda.

5. Membangun dan menjalankan kontainer

   Bangun dan jalankan wadah menggunakan Docker Compose:

   ```
   docker-compose up --build
   ```

6. Langkah selanjutnya

   Anda sekarang dapat menggunakan Penyedia Kre AWS denSIAL Beban Kerja untuk mengambil rahasia dari wadah klien Anda. Untuk informasi selengkapnya, lihat the section called “Mengambil rahasia dengan AWS Penyedia Kredensia Beban Kerja”.

Lambda

Anda dapat mengemas Penyedia Kreden AWS SIAL Beban Kerja sebagai ekstensi Lambda. Kemudian Anda dapat men ambahkannya ke fungsi Lambda Anda sebagai lapisan dan memanggil Penyedia KredenSIAL AWS Beban Kerja dari fungsi Lambda Anda untuk mendapatkan rahasia.

Petunjuk berikut menunjukkan cara mendapatkan nama rahasia MyTest dengan menggunakan skrip contoh `secrets-manager-provider-extension.sh` di GitHub repositori aws-workload-credentials-provider untuk menginstal Penyedia KredenSIAL Beban Kerja sebagai ekstensi Lambda. AWS

Untuk membuat ekstensi Lambda untuk AWS Penyedia Kredensia Beban Kerja

1. Kemas lapisan penyedia

   Dari root paket kode Penyedia AWS Kredentifikasi Beban Kerja, jalankan perintah berikut:

```
AWS_ACCOUNT_ID=AWS_ACCOUNT_ID
LAMBDA_ARN=LAMBDA_ARN

# Build the release binary
cargo build --release --target=x86_64-unknown-linux-gnu

# Copy the release binary into the `bin` folder
mkdir -p ./bin
cp ./target/x86_64-unknown-linux-gnu/release/aws-workload-credentials-
provider ./bin/aws-workload-credentials-provider

# Copy the `secrets-manager-provider-extension.sh` example script into the
 `extensions` folder.
mkdir -p ./extensions
cp aws_secretsmanager_provider/examples/example-lambda-extension/secrets-
manager-provider-extension.sh ./extensions

# Zip the extension shell script and the binary
zip secrets-manager-provider-extension.zip bin/* extensions/*

# Publish the layer version
LAYER_VERSION_ARN=$(aws lambda publish-layer-version \
    --layer-name secrets-manager-provider-extension \
    --zip-file "fileb://secrets-manager-provider-extension.zip" | jq -r
 '.LayerVersionArn')
```

2. Konfigurasikan token SSRF

   Konfigurasi default penyedia akan secara otomatis mengatur token SSRF ke nilai yang ditetapkan dalam variabel pra-set `AWS_SESSION_TOKEN` atau `AWS_CONTAINER_AUTHORIZATION_TOKEN` lingkungan (variabel terakhir untuk fungsi Lambda dengan SnapStart diaktifkan). Atau, Anda dapat menentukan variabel `AWS_TOKEN` lingkungan dengan nilai arbitrer untuk fungsi Lambda Anda sebagai gantinya karena variabel ini diutamakan daripada dua lainnya. Jika Anda memilih untuk menggunakan variabel `AWS_TOKEN` lingkungan, Anda harus mengatur variabel lingkungan itu dengan `lambda:UpdateFunctionConfiguration` panggilan.

3. Pasang lapisan ke fungsi

   Lampirkan versi layer ke fungsi Lambda Anda:

   ```
   # Attach the layer version to the Lambda function
   aws lambda update-function-configuration \
       --function-name $LAMBDA_ARN \
       --layers "$LAYER_VERSION_ARN"
   ```

4. Perbarui kode fungsi

   Perbarui fungsi Lambda Anda untuk men `http://localhost:2773/secretsmanager/get?secretId=MyTest` anyakan dengan nilai `X-Aws-codes-Secrets-Token` header yang disetel ke nilai token SSRF yang bersumber dari salah satu variabel lingkungan yang disebutkan di atas untuk mengambil rahasia. Pastikan untuk menerapkan logika coba ulang dalam kode aplikasi Anda untuk mengakomodasi penundaan inisialisasi dan pendaftaran ekstensi Lambda.

5. Uji fungsi

   Panggil fungsi Lambda untuk memverifikasi bahwa rahasia sedang diambil dengan benar.

Mengambil rahasia dengan AWS Penyedia Kredensia Beban Kerja

Untuk mengambil rahasia, panggil titik akhir Penyedia Kre AWS dentifikasi Beban Kerja lokal dengan nama rahasia atau ARN sebagai parameter kueri. Secara default, Penyedia AWS KredenSIAL Beban Kerja `AWSCURRENT` mengambil versi rahasia. Untuk mengambil versi yang berbeda, gunakan parameter versionStage atau versionId.

> **Important**
>
> Untuk membantu melindungi Penyedia AWS Kredensi Beban Kerja, Anda harus menyertakan header token SSRF sebagai bagian dari setiap permintaan:`X-Aws-Parameters-Secrets-Token`. Penye AWS dia Kredensi Beban Kerja menolak permintaan yang tidak memiliki header ini atau yang memiliki token SSRF yang tidak valid. Anda dapat menyesuaikan nama header SSRF dithe section called “Opsi konfigurasi”.

Izin yang diperlukan

Penyedia AWS Kredensi Beban AWS Kerja menggunakan SDK untuk Rust, yang menggunakan rantai penyedia AWS kredensia. Identitas kredentif IAM ini menentukan izin yang dimiliki Penyedia Kreden AWS SIAL Beban Kerja untuk mengambil rahasia.

- `secretsmanager:DescribeSecret`
- `secretsmanager:GetSecretValue`

Untuk informasi selengkapnya tentang izin, lihat the section called “Referensi izin”.

> **Important**
>
> Setelah nilai rahasia ditarik ke Penyedia KredenSIAL AWS Beban Kerja, setiap pengguna dengan akses ke lingkungan komputasi dan token SSRF dapat mengakses rahasia dari cache Penyedia KredenSIAL B AWS eban Kerja. Untuk informasi selengkapnya, lihat the section called “Pertimbangan keamanan”.

Contoh permintaan

curl

Example Contoh — Dapatkan rahasia menggunakan curl

Contoh curl berikut menunjukkan cara mendapatkan rahasia dari Penyedia KredenSIAL B AWS eban Kerja. Contohnya bergantung pada SSRF yang ada dalam file, di mana ia disimpan oleh skrip instal.

```
curl -v -H \
    "X-Aws-Parameters-Secrets-Token: $(</var/run/awssmatoken)" \
    'http://localhost:2773/secretsmanager/get?secretId=YOUR_SECRET_ID'
```

Python

Example Contoh — Dapatkan rahasia menggunakan Python

Contoh Python berikut menunjukkan cara mendapatkan rahasia dari Penyedia KredenSIAL B AWS eban Kerja. Contohnya bergantung pada SSRF yang ada dalam file, di mana ia disimpan oleh skrip instal.

```
import requests
import json

# Function that fetches the secret from AWS Workload Credentials Provider for the
 provided secret id.
def get_secret():
    # Construct the URL for the GET request
    url = f"http://localhost:2773/secretsmanager/get?secretId=YOUR_SECRET_ID"

    # Get the SSRF token from the token file
    with open('/var/run/awssmatoken') as fp:
        token = fp.read()

    headers = {
        "X-Aws-Parameters-Secrets-Token": token.strip()
    }

    try:
        # Send the GET request with headers
        response = requests.get(url, headers=headers)

        # Check if the request was successful
        if response.status_code == 200:
            # Return the secret value
            return response.text
        else:
            # Handle error cases
            raise Exception(f"Status code {response.status_code} - {response.text}")

    except Exception as e:
        # Handle network errors
        raise Exception(f"Error: {e}")
```

Memahami parameter **`RefreshNow`**

Penyedia AWS Kredensi Beban Kerja menggunakan cache dalam memori untuk menyimpan nilai rahasia, yang disegarkan secara berkala. Secara default, penyegaran ini terjadi ketika Anda meminta rahasia setelah Time to Live (TTL) berakhir, biasanya setiap 300 detik. Namun, pendekatan ini terkadang dapat mengakibatkan nilai rahasia basi, terutama jika rahasia berputar sebelum entri cache berakhir.

Untuk mengatasi batasan ini, Penyedia Kre AWS densi Beban Kerja mendukung parameter yang di `refreshNow` panggil di URL. Anda dapat menggunakan parameter ini untuk memaksa penyegaran langsung nilai rahasia, melewati cache dan memastikan Anda memiliki informasi terbaru.

Perilaku default (tanpa**`refreshNow`**)

- Menggunakan nilai yang di-cache hingga TTL kedaluwarsa
- Menyegarkan rahasia hanya setelah TTL (default 300 detik)
- Dapat mengembalikan nilai basi jika rahasia berputar sebelum cache berakhir

Perilaku dengan **`refreshNow=true`**

- Melewati cache sepenuhnya
- Mengambil nilai rahasia terbaru langsung dari Secrets Manager
- Memperbarui cache dengan nilai baru dan mengatur ulang TTL
- Memastikan Anda selalu mendapatkan nilai rahasia terbaru

Force-refresh nilai rahasia

> **Important**
>
> Nilai default `refreshNow` adalah `false`. Ketika disetel ke`true`, itu mengganti TTL yang ditentukan dalam file konfigurasi Penyedia Kre AWS denSIAL Beban Kerja dan membuat panggilan API ke Manajer Rahasia.

curl

Example Contoh — Force-refresh rahasia menggunakan curl

Contoh curl berikut menunjukkan cara memaksa Penyedia KredenSIAL AWS Beban Kerja untuk menyegarkan rahasia. Contohnya bergantung pada SSRF yang ada dalam file, di mana ia disimpan oleh skrip instal.

```
curl -v -H \
"X-Aws-Parameters-Secrets-Token: $(</var/run/awssmatoken)" \
'http://localhost:2773/secretsmanager/get?secretId=YOUR_SECRET_ID&refreshNow=true'
```

Python

Example Contoh — Force-refresh rahasia menggunakan Python

Contoh Python berikut menunjukkan cara mendapatkan rahasia dari Penyedia KredenSIAL B AWS eban Kerja. Contohnya bergantung pada SSRF yang ada dalam file, di mana ia disimpan oleh skrip instal.

```
import requests
import json

# Function that fetches the secret from AWS Workload Credentials Provider for the
 provided secret id.
def get_secret():
    # Construct the URL for the GET request
    url = f"http://localhost:2773/secretsmanager/get?
secretId=YOUR_SECRET_ID&refreshNow=true"

    # Get the SSRF token from the token file
    with open('/var/run/awssmatoken') as fp:
        token = fp.read()

    headers = {
        "X-Aws-Parameters-Secrets-Token": token.strip()
    }

    try:
        # Send the GET request with headers
        response = requests.get(url, headers=headers)

        # Check if the request was successful
        if response.status_code == 200:
            # Return the secret value
            return response.text
        else:
            # Handle error cases
            raise Exception(f"Status code {response.status_code} - {response.text}")

    except Exception as e:
        # Handle network errors
        raise Exception(f"Error: {e}")
```

Mengambil rahasia di seluruh akun dengan rantai peran

Rantaian peran memungkinkan Penyedia Kre AWS densi Beban Kerja untuk mengambil rahasia dari AWS akun lain dengan mengasumsikan peran IAM menggunakan. AWS STS `AssumeRole` Penyedia AWS Kredensi Beban Kerja membuat dan menyimpan klien caching terpisah untuk setiap peran unik ARN. Setiap klien peran mempertahankan cache independennya sendiri, sehingga rahasia yang sama diambil dengan peran yang berbeda memiliki entri cache terpisah.

Izin yang diperlukan

Untuk menggunakan rantai peran, Anda memerlukan yang berikut:

- Kreden AWS tif lingkungan Penyedia Kredensia Beban Kerja harus memiliki `sts:AssumeRole` izin pada peran target ARN.
- Peran target harus memiliki `secretsmanager:GetSecretValue` dan `secretsmanager:DescribeSecret` izin untuk rahasia yang ingin Anda akses.
- Kebijakan kepercayaan peran target harus mengizinkan identitas Penyedia Kre AWS dentif Beban Kerja untuk mengasumsinya.

Mengambil rahasia lintas akun

Sertakan parameter `roleArn` kueri dalam permintaan Anda ke Penyedia Kre AWS densi Beban Kerja untuk menentukan peran mana yang akan diambil untuk pengambilan rahasia.

curl

Example Contoh — Cross-account rahasia menggunakan curl

```
curl -v -H \
    "X-Aws-Parameters-Secrets-Token: $(</var/run/awssmatoken)" \
    'http://localhost:2773/secretsmanager/get?
secretId=YOUR_SECRET_ID&roleArn=arn:aws:iam::ACCOUNT_ID:role/ROLE_NAME'
```

Python

Example Contoh — Cross-account rahasia menggunakan Python

```
import requests

def get_secret_cross_account():
```

```
    secret_id = "YOUR_SECRET_ID"
    role_arn = "arn:aws:iam::ACCOUNT_ID:role/ROLE_NAME"
    url = f"http://localhost:2773/secretsmanager/get?
secretId={secret_id}&roleArn={role_arn}"

    with open('/var/run/awssmatoken') as fp:
        token = fp.read()

    headers = {
        "X-Aws-Parameters-Secrets-Token": token.strip()
    }

    try:
        response = requests.get(url, headers=headers)

        if response.status_code == 200:
            return response.text
        else:
            raise Exception(f"Status code {response.status_code} - {response.text}")

    except Exception as e:
        raise Exception(f"Error: {e}")
```

Konfigurasi dan batasan rantai peran

Konfigurasikan rantai peran dengan `max_roles` opsi di file konfigurasi TOML Anda. Ini menetapkan jumlah maksimum peran yang diasumsikan secara simultan, dalam kisaran 1 hingga 20. Defaultnya adalah 20.

> **Important**
>
> Peran yang diasumsikan tidak diusir dari cache peran Penyedia AWS Kredensi Beban Kerja. Setelah jumlah maksimum peran tercapai, permintaan dengan ARN peran baru ditolak dengan `400` kesalahan hingga Penyedia KredenSIAL Beb AWS an Kerja dimulai ulang.

Tanggapan kesalahan untuk rantai peran

`400`

`roleArn`Format tidak valid atau jumlah maksimum peran yang diasumsikan telah tercapai.

403

Pang AWS STS `AssumeRole` gilan gagal. Verifikasi bahwa kebijakan kepercayaan peran target memungkinkan identitas Penyedia Kreden AWS tif Beban Kerja untuk mengasumsikannya.

Pre-fetch rahasia saat startup

Secara default, Penyedia Kre AWS densi Beban Kerja mengambil rahasia sesuai permintaan saat aplikasi Anda memintanya. Dengan pra-pengambilan, Penyedia AWS KredenSIAL Beban Kerja memuat rahasia yang ditentukan ke dalam cache saat dijalankan, sehingga aplikasi Anda dapat langsung mengaksesnya tanpa menunggu panggilan API pertama. Pre-fetching berjalan sebagai tugas latar belakang—Penyedia Kre AWS denSIAL Beban Kerja mulai menerima permintaan segera dan tidak memblokir penyelesaian pra-pengambilan.

Anda dapat menentukan rahasia untuk pra-pengambilan dengan dua cara:

- Rahasia eksplisit — Daftar ID rahasia tertentu atau ARN.
- Tag-based penemuan — Temukan rahasia dengan kunci tag. Penyedia AWS KredenSIAL Beban Kerja mengambil semua rahasia yang memiliki tag yang ditentukan.

Izin yang diperlukan

Selain izin standar untuk mengambil rahasia, pra-pengambilan memerlukan hal berikut:

- `secretsmanager:BatchGetSecretValue`— Diperlukan untuk semua operasi pra-fetch.
- `secretsmanager:ListSecrets`— Diperlukan hanya saat menggunakan penemuan berbasis tag.

Konfigurasikan pra-pengambilan

Tambahkan `[capabilities.secrets_manager.prefetch]` bagian ke file konfigurasi TOML Anda. Pilihan berikut tersedia:

`cache_buffer_ratio`

Fraksi maksimum cache untuk diisi per klien selama pra-fetch, dalam kisaran 0,1 hingga 1,0. Defaultnya adalah 0,8. Ketika batas buffer tercapai, Penyedia Kredensi B AWS eban Kerja

menghentikan pra-pengambilan rahasia yang tersisa — tidak menghapus entri cache yang ada. Rahasia yang tidak dimuat selama pra-pengambilan masih tersedia sesuai permintaan.

`max_jitter_seconds`

Penundaan acak dalam detik sebelum pra-pengambilan dimulai, dalam kisaran 0 hingga 10. Default-nya adalah 0. Gunakan ini untuk mencegah panggilan API seluruh armada yang disinkronkan saat beberapa penyedia memulai secara bersamaan.

Example Pre-fetch konfigurasi dengan rahasia eksplisit

```
[capabilities.secrets_manager.prefetch]
cache_buffer_ratio = 0.6
max_jitter_seconds = 5
secrets = [
    { secret_id = "arn:aws:secretsmanager:us-west-2:123456789012:secret:MySecret-
AbCdEf" },
    { secret_id = "MyOtherSecret" },
]
```

Example Pre-fetch konfigurasi dengan penemuan berbasis tag

```
[capabilities.secrets_manager.prefetch]
cache_buffer_ratio = 0.8
filter_tags = [
    { key = "Environment" },
    { key = "Team" },
]
```

Anda juga dapat menggabungkan rahasia eksplisit dan penemuan berbasis tag dalam konfigurasi yang sama. Untuk pra-pengambilan lintas akun, tambahkan bidang. `role_arn` Untuk informasi selengkapnya, lihat the section called “Mengambil rahasia di seluruh akun dengan rantai peran”.

Example Pre-fetch konfigurasi dengan akses lintas akun

```
[capabilities.secrets_manager.prefetch]
cache_buffer_ratio = 0.6
max_jitter_seconds = 5
secrets = [
    { secret_id = "arn:aws:secretsmanager:us-west-2:123456789012:secret:MySecret-
AbCdEf" },
```

```
    { secret_id = "cross-account-secret", role_arn = "arn:aws:iam::987654321098:role/
SecretAccessRole" },
]
filter_tags = [
    { key = "Environment" },
    { key = "Team", role_arn = "arn:aws:iam::987654321098:role/SecretAccessRole" },
]
```

Konfigurasikan AWS Penyedia Kredensia Beban Kerja

Untuk mengubah konfigurasi Penyedia Kreden AWS SIAL Beban Kerja, buat file konfigurasi TOML, lalu panggil `./aws-workload-credentials-provider sm start --config config.toml`.

File konfigurasi mendukung format bersarang. Opsi Manajer Rahasia ditempatkan di bawah `[capabilities.secrets_manager]`, dengan sub-bagian untuk cache dan pengaturan keamanan. Opsi pencatatan ditempatkan di bawah `[logging]`.

Example Contoh file konfigurasi bersarang

```
[logging]
log_level = "INFO"
log_to_file = true

[capabilities.secrets_manager]
enabled = true
http_port = 2773
region = "us-east-1"
path_prefix = "/v1/"
max_conn = 800
max_roles = 20

[capabilities.secrets_manager.cache]
ttl_seconds = 300
cache_size = 1000

[capabilities.secrets_manager.security]
ssrf_headers = ["X-Aws-Parameters-Secrets-Token", "X-Vault-Token"]
ssrf_env_variables = ["AWS_TOKEN", "AWS_SESSION_TOKEN",
 "AWS_CONTAINER_AUTHORIZATION_TOKEN"]
```

> **Note**
>
> Tombol datar di tingkat root (misalnya,`http_port = 2773`) masih didukung untuk kompatibilitas mundur dengan file konfigurasi yang ada.

Opsi konfigurasi Manajer Rahasia

`enabled`

Apakah kemampuan Manajer Rahasia aktif: `true` atau`false`. Nilai default-nya `true`.

`http_port`

Port untuk server HTTP lokal, dalam kisaran 1024 hingga 65535. Defaultnya adalah 2773.

`region`

Wil AWS ayah yang akan digunakan untuk permintaan. Jika tidak ada Wilayah yang ditentukan, Penyedia AWS Kredensi Beban Kerja menentukan Wilayah dari SDK. Untuk informasi selengkapnya, lihat Menentukan kredensi dan Wilayah default Anda di AWS SDK untuk Panduan Pengembang Rust.

`path_prefix`

Awalan URI digunakan untuk menentukan apakah permintaan adalah permintaan berbasis jalur. Defaultnya adalah “/v1/”.

`max_conn`

Jumlah maksimum koneksi dari klien HTTP yang diizinkan oleh Penyedia Kre AWS densi Beban Kerja, dalam kisaran 1 hingga 1000. Defaultnya adalah 800.

`max_roles`

Jumlah maksimum peran IAM simultan untuk akses lintas akun, dalam kisaran 1 hingga 20. Defaultnya adalah 20. Untuk informasi selengkapnya, lihat the section called “Mengambil rahasia di seluruh akun dengan rantai peran”.

Opsi cache (**`[capabilities.secrets_manager.cache])`**

`ttl_seconds`

TTL dalam hitungan detik untuk item yang di-cache, dalam kisaran 0 hingga 3600. Defaultnya adalah 300. 0 menunjukkan bahwa tidak ada caching.

`cache_size`

Jumlah maksimum rahasia yang dapat disimpan dalam cache, dalam kisaran 1 hingga 1000. Defaultnya adalah 1000.

Opsi keamanan (**`[capabilities.secrets_manager.security])`**

`ssrf_headers`

Daftar nama header yang diperiksa oleh Penyedia AWS KredenSIAL Beban Kerja untuk token SSRF. Defaultnya adalah "X-Aws-Parameters-Secrets-Token, X-Vault-Token”.

`ssrf_env_variables`

Daftar nama variabel lingkungan yang diperiksa Penyedia AWS KredenSIAL Beban Kerja dalam urutan berurutan untuk token SSRF. Variabel lingkungan dapat berisi token atau referensi ke file token seperti di:`AWS_TOKEN=file:///var/run/awssmatoken`. Defaultnya adalah "AWS_TOKEN, AWS_SESSION_TOKEN, AWS_CONTAINER_AUTHORIZATION_TOKEN”.

Opsi pencatatan (**`[logging]`**)

`log_level`

Tingkat detail yang dilaporkan dalam log untuk Penyedia KredenSIAL B AWS eban Kerja: DEBUG, INFO, WARN, ERROR, atau NONE. Defaultnya adalah INFO.

`log_to_file`

Apakah akan masuk ke file atau stdout/stderr: `true` atau`false`. Nilai default-nya `true`.

Fitur opsional

Penyedia AWS Kredensi Beban Kerja dapat dibangun dengan fitur opsional dengan meneruskan `--features` bendera `kecargo build`. Fitur yang tersedia adalah:

Membangun fitur

`prefer-post-quantum`

Membuat `X25519MLKEM768` algoritma pertukaran kunci prioritas tertinggi. Jika tidak, tersedia tetapi bukan prioritas tertinggi. `X25519MLKEM768`adalah algoritma pertukaran kunci pasca-kuantum hibrida yang aman.

`fips`

Membatasi suite sandi yang digunakan oleh penyedia hanya FIPS-approved untuk cipher.

Pencatatan log

Pencatatan lokal

Penye AWS dia KredenSIAL Beban Kerja mencatat kesalahan secara lokal ke file `logs/secrets_manager_provider.log` atau stdout/stderr tergantung pada variabel `log_to_file` konfigurasi. Saat aplikasi Anda memanggil Penye AWS dia Kredensi Beban Kerja untuk mendapatkan rahasia, panggilan tersebut muncul di log lokal. Mereka tidak muncul di CloudTrail log.

Rotasi log

Penye AWS dia Kredensi Beban Kerja membuat file log baru ketika file mencapai 10 MB, dan menyimpan hingga lima file log total.

AWS pencatatan layanan

Log tidak masuk ke Manajer Rahasia, CloudTrail, atau CloudWatch. Permintaan untuk mendapatkan rahasia dari Penyedia AWS Kredensi Beban Kerja tidak muncul di log tersebut. Ketika Penye AWS dia Kredensia Beban Kerja membuat panggilan ke Manajer Rahasia untuk mendapatkan rahasia, panggilan itu direkam CloudTrail dengan string agen pengguna yang berisi`aws-workload-credentials-provider`.

Anda dapat mengonfigurasi opsi logging dithe section called “Opsi konfigurasi”.

Pertimbangan keamanan

Domain kepercayaan

Untuk arsitektur penyedia lokal, domain kepercayaan adalah tempat titik akhir penyedia dan token SSRF dapat diakses, yang biasanya merupakan seluruh host. Domain kepercayaan untuk Penyedia Kre AWS denSIAL Beban Kerja harus sesuai dengan domain tempat kredenSIAL Manajer Rahasia tersedia untuk mempertahankan postur keamanan yang sama. Misalnya, di Amazon EC2 domain kepercayaan untuk Penyedia Kreden AWS SIAL Beban Kerja akan sama dengan domain kredenSIAL saat menggunakan peran untuk Amazon EC2.

Important

Aplikasi sadar keamanan yang belum menggunakan solusi berbasis penyedia dengan kredenSIAL Manajer Rahasia yang terkunci ke aplikasi harus mempertimbangkan untuk menggunakan AWS SDK khusus bahasa atau solusi caching. Untuk informasi selengkapnya, lihat M endapatkan rahasia.

Gunakan rahasia dengan aman dengan Agen Pengkodean AI

Ketika agen pengkodean AI memiliki akses shell atau AWS API, mereka dapat menelepon `get-secret-value` dan menerima rahasia teks biasa di jendela konteks mereka. Ini menciptakan banyak risiko: nilai rahasia dapat bocor ke riwayat percakapan, log, atau panggilan alat hilir.

Untuk mencegah hal ini, gunakan keterampilan keamanan rahasia di Agen Toolkit untuk AWS. Keterampilan ini mengajarkan agen AI untuk menggunakan referensi dinamis yang diselesaikan saat runtime, sehingga agen mengatur penggunaan rahasia tanpa pernah melihat nilai plaintext.

Important

Ini adalah pertahanan upaya terbaik, bukan batas keamanan. Ini mencegah jalur kebocoran yang paling umum tetapi tidak dapat menghentikan semua vektor penghindaran. Kombinasikan dengan kebijakan hak istimewa, pemantauan, dan titik akhir VPC CloudTrail IAM.

Cara kerjanya

Keterampilan keamanan rahasia memberikan dua lapisan perlindungan:

1. Panduan keterampilan — Mengajarkan agen untuk menggunakan referensi `{{resolve:secretsmanager:...}}` dinamis dengan`asm-exec`, skrip pembungkus yang menyelesaikan referensi saat runtime. Nilai plaintext hanya ada dalam proses anak dan tidak pernah memasuki jendela konteks agen.
2. Penegakan struktural (hook) - `PreToolUse` Hook secara otomatis memblokir setiap upaya untuk memanggil `get-secret-value` atau `batch-get-secret-value` melalui AWS CLI, SDK, alat MCP, atau akses langsung ke daemon Penyedia Kredensyal AWS Beban Kerja. Tidak diperlukan konfigurasi manual.

Prasyarat

- Agen pengkodean AI yang mendukung plugin, seperti Claude Code atau OpenAI Codex.
- Agen Toolkit untuk AWS `aws-core` plugin diinstal.
- Salah satu backend resolusi rahasia berikut:
  - AWS Penyedia Kredensial Beban Kerja berjalan. `localhost:2773` Lihat the section called “AWS Penyedia Kredensia Beban Kerja”.
  - AWS kredensyal yang dapat menandatangani permintaan ke titik akhir AWS MCP.
- Izin IAM: `secretsmanager:GetSecretValue` pada rahasia yang ingin Anda selesaikan.

Pasang plugin

Instal `aws-core` plugin untuk platform agen Anda. Keterampilan keamanan rahasia dan kait diaktifkan secara otomatis.

Untuk Kode Claude:

```
claude plugin add ./plugins/aws-core
```

Untuk OpenAI Codex:

```
codex plugin add ./plugins/aws-core
```

Untuk platform lain yang didukung, lihat Toolkit Agen untuk AWS README.

{{resolve:...}} sintaksis

Ketika agen perlu meneruskan rahasia ke perintah, ia menggunakan referensi dinamis alih-alih memanggil`get-secret-value`:

```
{{resolve:secretsmanager:<secret-id>:<field-type>:<json-key>:<version-stage>}}
```

Komponen	Diperlukan	Default	Deskripsi
`secret-id`	Ya	–	Nama rahasia atau ARN lengkap
`field-type`	Tidak	`SecretString`	Harus `SecretString`
`json-key`	Tidak	(nilai penuh)	Kunci untuk mengekstrak dari nilai rahasia JSON
`version-stage`	Tidak	`AWSCURRENT`	Label tahap versi

Gunakan **`asm-exec`** untuk menjalankan perintah dengan rahasia

`asm-exec`adalah skrip pembungkus yang menyelesaikan `{{resolve:...}}` referensi dalam argumen perintah, lalu menjalankan perintah target. Nilai rahasia hanya ada dalam proses anak.

```
asm-exec -- <command> [arguments with {{resolve:...}} references]
```

`asm-exec`menyelesaikan referensi melalui backend pertama yang tersedia:

1. AWS Penyedia Kredensial Beban Kerja aktif **`localhost:2773`** — di-cache secara lokal.
2. AWS MCP endpoint — SigV4-signed permintaan menggunakan kredensyal yang tersedia AWS.

Example Connect ke database PostgreSQL

```
asm-exec -- psql \
```

```
  "host=mydb.example.com \
  user={{resolve:secretsmanager:prod/db-creds:SecretString:username}} \
  password={{resolve:secretsmanager:prod/db-creds:SecretString:password}}" \
  -c "SELECT * FROM users LIMIT 10"
```

Example Lakukan panggilan API dengan token pembawa

```
asm-exec -- curl -H "Authorization: Bearer {{resolve:secretsmanager:prod/api-token}}" \
  https://api.example.com/data
```

Example Connect ke MySQL dengan beberapa rahasia

```
asm-exec -- mysql \
  -h {{resolve:secretsmanager:prod/mysql:SecretString:host}} \
  -u {{resolve:secretsmanager:prod/mysql:SecretString:username}} \
  -p{{resolve:secretsmanager:prod/mysql:SecretString:password}} \
  -e "SHOW TABLES"
```

Example Berikan rahasia sebagai variabel lingkungan ke wadah Docker

```
asm-exec -- docker run \
  -e "DB_PASSWORD={{resolve:secretsmanager:prod/db:SecretString:password}}" \
  myapp:latest
```

Cross-region rahasia

Untuk rahasia yang disimpan di wilayah yang berbeda dari wilayah default Anda, gunakan ARN lengkap (yang mencakup wilayah) atau setel variabel `AWS_REGION` lingkungan.

```
# Using full ARN (region is extracted automatically)
asm-exec -- curl -H "X-Api-Key: {{resolve:secretsmanager:arn:aws:secretsmanager:eu-
west-1:123456789012:secret:prod/key-a1b2c3}}" \
  https://eu.api.example.com/data

# Using AWS_REGION
export AWS_REGION=eu-west-1
asm-exec -- curl -H "X-Api-Key: {{resolve:secretsmanager:prod/key}}" \
  https://eu.api.example.com/data
```

Pertimbangan keamanan

- Isolasi subproses — Perintah target berjalan melaluisubprocess.run. Nilai rahasia hanya ada dalam memori asm-exec proses dan argumen proses anak.

Bagaimana hook memblokir akses rahasia langsung

Ketika aws-core plugin diaktifkan, PreToolUse hook mencegat panggilan alat sebelum eksekusi. Ini memblokir:

- aws secretsmanager get-secret-valuedan batch-get-secret-value melalui CLI
- get_secret_valuedan batch_get_secret_value melalui panggilan SDK dalam skrip
- Akses langsung ke jalur daemon Penyedia Kredensial AWS Beban Kerja () localhost:2773/secretsmanager/get
- GetSecretValueoperasi melalui alat MCP atau panggilan AWS API terstruktur

Ketika panggilan diblokir, agen menerima pesan penolakan yang mengarahkannya untuk menggunakan {{resolve:...}} referensi asm-exec sebagai gantinya.

Pemecahan masalah

Kesalahan “Rahasia tidak ditemukan”

Verifikasi rahasia itu ada dan peran IAM Anda memiliki secretsmanager:GetSecretValue izin. Nama rahasia peka huruf besar/kecil.

AWS Beban Kerja Kredensyal Koneksi penyedia ditolak

Penyedia Kredensyal AWS Beban Kerja mungkin tidak berjalan. Ini tidak fatal — asm-exec jatuh ke titik akhir SigV4-signed MCP. Pastikan AWS kredensyal tersedia sehingga backend dapat mengautentikasi.

Kesalahan “Gagal menyelesaikan”

Kedua backend tidak dapat dijangkau. Periksa apakah Penyedia Kredensyal AWS Beban Kerja sedang berjalan atau AWS kredensialnya valid (aws sts get-caller-identity), bahwa wilayah rahasia sudah benar, dan identitas Anda ada di rahasia tersebut. secretsmanager:GetSecretValue

Resolusi menghasilkan string kosong

Kunci JSON mungkin tidak ada dalam nilai rahasia. Verifikasi struktur rahasia di AWS konsol atau minta pemilik rahasia untuk mengonfirmasi kunci yang tersedia.

Hook tidak memblokir panggilan

Kait dimuat saat sesi agen dimulai. Jika Anda menginstal plugin pertengahan sesi, mulai ulang sesi agen agar hook dapat diaktifkan.

Dapatkan nilai rahasia Manajer Rahasia menggunakan C ++ AWS SDK

Untuk aplikasi C++, panggil SDK secara langsung dengan GetSecretValue atau BatchGetSecretValue.

Contoh kode berikut menunjukkan cara mendapatkan nilai rahasia Manajer Rahasia.

Izin yang diperlukan: `secretsmanager:GetSecretValue`

```
//! Retrieve an AWS Secrets Manager encrypted secret.
/*!
  \param secretID: The ID for the secret.
  \return bool: Function succeeded.
 */
bool AwsDoc::SecretsManager::getSecretValue(const Aws::String &secretID,
                                            const Aws::Client::ClientConfiguration
 &clientConfiguration) {
    Aws::SecretsManager::SecretsManagerClient
 secretsManagerClient(clientConfiguration);

    Aws::SecretsManager::Model::GetSecretValueRequest request;
    request.SetSecretId(secretID);

    Aws::SecretsManager::Model::GetSecretValueOutcome getSecretValueOutcome =
 secretsManagerClient.GetSecretValue(
            request);
    if (getSecretValueOutcome.IsSuccess()) {
        std::cout << "Secret is: "
                  << getSecretValueOutcome.GetResult().GetSecretString() << std::endl;
    }
```

```
    else {
        std::cerr << "Failed with Error: " << getSecretValueOutcome.GetError()
                  << std::endl;
    }

    return getSecretValueOutcome.IsSuccess();
}
```

Dapatkan nilai rahasia Manajer Rahasia menggunakan JavaScript AWS SDK

Untuk JavaScript aplikasi, hubungi SDK secara langsung dengan `getSecretValue` atau `batchGetSecretValue`.

Contoh kode berikut menunjukkan cara mendapatkan nilai rahasia Manajer Rahasia.

Izin yang diperlukan: `secretsmanager:GetSecretValue`

```
import {
  GetSecretValueCommand,
  SecretsManagerClient,
} from "@aws-sdk/client-secrets-manager";

export const getSecretValue = async (secretName = "SECRET_NAME") => {
  const client = new SecretsManagerClient();
  const response = await client.send(
    new GetSecretValueCommand({
      SecretId: secretName,
    }),
  );
  console.log(response);
  // {
  //   '$metadata': {
  //     httpStatusCode: 200,
  //     requestId: '584eb612-f8b0-48c9-855e-6d246461b604',
  //     extendedRequestId: undefined,
  //     cfId: undefined,
  //     attempts: 1,
  //     totalRetryDelay: 0
  //   },
  //   ARN: 'arn:aws:secretsmanager:us-east-1:xxxxxxxxxxxx:secret:binary-
secret-3873048-xxxxxx',
```

```
  //   CreatedDate: 2023-08-08T19:29:51.294Z,
  //   Name: 'binary-secret-3873048',
  //   SecretBinary: Uint8Array(11) [
  //      98, 105, 110, 97, 114,
  //     121,  32, 100, 97, 116,
  //      97
  //   ],
  //   VersionId: '712083f4-0d26-415e-8044-16735142cd6a',
  //   VersionStages: [ 'AWSCURRENT' ]
  // }

  if (response.SecretString) {
    return response.SecretString;
  }

  if (response.SecretBinary) {
    return response.SecretBinary;
  }
};
```

Dapatkan nilai rahasia Manajer Rahasia menggunakan Kotlin AWS SDK

Untuk aplikasi Kotlin, hubungi SDK secara langsung dengan GetSecretValue atau BatchGetSecretValue.

Contoh kode berikut menunjukkan cara mendapatkan nilai rahasia Manajer Rahasia.

Izin yang diperlukan: `secretsmanager:GetSecretValue`

```
suspend fun getValue(secretName: String?) {
    val valueRequest =
        GetSecretValueRequest {
            secretId = secretName
        }

    SecretsManagerClient.fromEnvironment { region = "us-east-1" }.use { secretsClient -
>
        val response = secretsClient.getSecretValue(valueRequest)
        val secret = response.secretString
        println("The secret value is $secret")
    }
```

```
}
```

Dapatkan nilai rahasia Manajer Rahasia menggunakan PHP AWS SDK

Untuk aplikasi PHP, hubungi SDK secara langsung dengan `GetSecretValue` atau `BatchGetSecretValue`.

Contoh kode berikut menunjukkan cara mendapatkan nilai rahasia Manajer Rahasia.

Izin yang diperlukan: `secretsmanager:GetSecretValue`

```
<?php

  /**
    * Use this code snippet in your app.
    *
    * If you need more information about configurations or implementing the sample
 code, visit the AWS docs:
    * https://aws.amazon.com/developer/language/php/
    */

  require 'vendor/autoload.php';

  use Aws\SecretsManager\SecretsManagerClient;
  use Aws\Exception\AwsException;

  /**
    * This code expects that you have AWS credentials set up per:
    * https://<<{{DocsDomain}}>>/sdk-for-php/v3/developer-guide/guide_credentials.html
    */

  // Create a Secrets Manager Client
  $client = new SecretsManagerClient([
      'profile' => 'default',
      'version' => '2017-10-17',
      'region' => '<<{{MyRegionName}}>>',
  ]);

  $secret_name = '<<{{MySecretName}}>>';
```

```
  try {
      $result = $client->getSecretValue([
          'SecretId' => $secret_name,
      ]);
  } catch (AwsException $e) {
      // For a list of exceptions thrown, see
      // https://<<{{DocsDomain}}>>/secretsmanager/latest/apireference/
API_GetSecretValue.html
      throw $e;
  }

  // Decrypts secret using the associated KMS key.
  $secret = $result['SecretString'];

  // Your code goes here
```

Dapatkan nilai rahasia Manajer Rahasia menggunakan Ruby AWS SDK

Untuk aplikasi Ruby, panggil SDK secara langsung dengan `get_secret_value` atau `batch_get_secret_value`.

Contoh kode berikut menunjukkan cara mendapatkan nilai rahasia Manajer Rahasia.

Izin yang diperlukan: `secretsmanager:GetSecretValue`

```
  # Use this code snippet in your app.
  # If you need more information about configurations or implementing the sample code,
 visit the AWS docs:
  # https://aws.amazon.com/developer/language/ruby/

  require 'aws-sdk-secretsmanager'

  def get_secret
    client = Aws::SecretsManager::Client.new(region: '<<{{MyRegionName}}>>')

    begin
      get_secret_value_response = client.get_secret_value(secret_id:
 '<<{{MySecretName}}>>')
    rescue StandardError => e
      # For a list of exceptions thrown, see
```

```
      # https://<<{{DocsDomain}}>>/secretsmanager/latest/apireference/
API_GetSecretValue.html
      raise e
    end

    secret = get_secret_value_response.secret_string
    # Your code goes here.
  end
```

Dapatkan nilai rahasia menggunakan AWS CLI

Izin yang diperlukan: `secretsmanager:GetSecretValue`

Example Ambil nilai rahasia terenkripsi dari rahasia

`get-secret-value`Contoh berikut mendapatkan nilai rahasia saat ini.

```
aws secretsmanager get-secret-value \
    --secret-id MyTestSecret
```

Example Ambil nilai rahasia sebelumnya

`get-secret-value`Contoh berikut mendapatkan nilai rahasia sebelumnya.

```
aws secretsmanager get-secret-value \
        --secret-id MyTestSecret
        --version-stage AWSPREVIOUS
```

Dapatkan sekelompok rahasia dalam satu batch menggunakan AWS CLI

Izin yang diperlukan:

- `secretsmanager:BatchGetSecretValue`
- `secretsmanager:GetSecretValue`izin untuk setiap rahasia yang ingin Anda ambil.
- Jika Anda menggunakan filter, Anda juga harus memilikinya`secretsmanager:ListSecrets`.

Untuk contoh kebijakan izin, lihatthe section called “Contoh: Izin untuk mengambil sekelompok nilai rahasia dalam batch”.

> **Important**
>
> Jika Anda memiliki kebijakan VPCE yang menolak izin untuk mengambil rahasia individu dalam grup yang Anda ambil, tidak `BatchGetSecretValue` akan mengembalikan nilai rahasia apa pun, dan itu akan mengembalikan kesalahan.

Example Ambil nilai rahasia untuk sekelompok rahasia yang terdaftar berdasarkan nama

`batch-get-secret-value`Contoh berikut mendapatkan nilai rahasia untuk tiga rahasia.

```
aws secretsmanager batch-get-secret-value \
          --secret-id-list MySecret1 MySecret2 MySecret3
```

Example Ambil nilai rahasia untuk sekelompok rahasia yang dipilih oleh filter

`batch-get-secret-value`Contoh berikut mendapatkan nilai rahasia untuk rahasia yang memiliki tag bernama “Test”.

```
aws secretsmanager batch-get-secret-value \
          --filters Key="tag-key",Values="Test"
```

Dapatkan nilai rahasia menggunakan AWS konsol

Untuk mengambil rahasia (konsol)

1. Buka konsol Secrets Manager di https://console.aws.amazon.com/secretsmanager/.
2. Dalam daftar rahasia, pilih rahasia yang ingin Anda ambil.
3. Di bagian Nilai rahasia, pilih Ambil nilai rahasia.

   Manajer Rahasia menampilkan versi rahasia saat ini (`AWSCURRENT`). Untuk melihat versi rahasia lainnya, seperti `AWSPREVIOUS` atau versi berlabel khusus, gunakanthe section called “AWS CLI”.

Gunakan AWS Secrets Manager rahasia di AWS Batch

AWS Batch membantu Anda menjalankan beban kerja komputasi batch di. AWS Cloud Dengan AWS Batch, Anda dapat menyuntikkan data sensitif ke dalam pekerjaan Anda dengan menyimpan data sensitif Anda dalam AWS Secrets Manager rahasia dan kemudian mereferensikannya dalam definisi

pekerjaan Anda. Untuk informasi selengkapnya, lihat Menentukan data sensitif menggunakan Secrets Manager.

Dapatkan AWS Secrets Manager rahasia di sumber CloudFormation daya

Dengan CloudFormation, Anda dapat mengambil rahasia untuk digunakan di CloudFormation sumber lain. Skenario umum adalah pertama-tama membuat rahasia dengan kata sandi yang dihasilkan oleh Secrets Manager, dan kemudian mengambil nama pengguna dan kata sandi dari rahasia untuk digunakan sebagai kredensyal untuk database baru. Untuk informasi tentang membuat rahasia dengan CloudFormation, lihatCloudFormation.

Untuk mengambil rahasia dalam CloudFormation template, Anda menggunakan referensi dinamis. Saat Anda membuat tumpukan, referensi dinamis menarik nilai rahasia ke CloudFormation sumber daya, jadi Anda tidak perlu melakukan hardcode informasi rahasia. Sebaliknya, Anda merujuk ke rahasia dengan nama atau ARN. Anda dapat menggunakan referensi dinamis untuk rahasia di properti sumber daya apa pun. Anda tidak dapat menggunakan referensi dinamis untuk rahasia dalam metadata sumber daya seperti `AWS::CloudFormation::Init`karena itu akan membuat nilai rahasia terlihat di konsol.

Referensi dinamis untuk rahasia memiliki pola berikut:

```
{{resolve:secretsmanager:secret-id:SecretString:json-key:version-stage:version-id}}
```

rahasia-id

Nama atau ARN rahasianya. Untuk mengakses rahasia di AWS akun Anda, Anda dapat menggunakan nama rahasia. Untuk mengakses rahasia di AWS akun yang berbeda, gunakan ARN rahasia.

json-key (Opsional)

Nama kunci dari pasangan kunci-nilai yang nilainya ingin Anda ambil. Jika Anda tidak menentukan`json-key`, CloudFormation mengambil seluruh teks rahasia. Segmen ini mungkin tidak memasukkan karakter titik dua (`:`).

tahap versi (Opsional)

Versi rahasia untuk digunakan. Secrets Manager menggunakan label pementasan untuk melacak versi yang berbeda selama proses rotasi. Jika Anda menggunakan `version-stage` maka

jangan tentukan `version-id`. Jika Anda tidak menentukan salah satu `version-stage` atau`version-id`, maka defaultnya adalah `AWSCURRENT` versinya. Segmen ini mungkin tidak memasukkan karakter titik dua (`:`).

version-id (Opsional)

Pengidentifikasi unik dari versi rahasia yang akan digunakan. Jika Anda menentukan `version-id`, jangan tentukan `version-stage`. Jika Anda tidak menentukan salah satu `version-stage` atau`version-id`, maka defaultnya adalah `AWSCURRENT` versinya. Segmen ini mungkin tidak memasukkan karakter titik dua (`:`).

Untuk informasi selengkapnya, lihat Menggunakan referensi dinamis untuk menentukan rahasia Secrets Manager.

> **Note**
>
> Jangan membuat referensi dinamis menggunakan garis miring terbalik (`\`) sebagai nilai akhir. CloudFormation tidak dapat menyelesaikan referensi tersebut, yang menyebabkan kegagalan sumber daya.

Gunakan AWS Secrets Manager rahasia dalam GitHub pekerjaan

Untuk menggunakan rahasia dalam GitHub pekerjaan, Anda dapat menggunakan GitHub tindakan untuk mengambil rahasia dari AWS Secrets Manager dan menambahkannya sebagai variabel Lingkungan bertopeng dalam alur kerja Anda GitHub . Untuk informasi selengkapnya tentang GitHub Tindakan, lihat Memahami GitHub Tindakan di GitHub Dokumen.

Ketika Anda menambahkan rahasia ke GitHub lingkungan Anda, itu tersedia untuk semua langkah lain dalam GitHub pekerjaan Anda. Ikuti panduan dalam Pengerasan Keamanan untuk GitHub Tindakan untuk membantu mencegah rahasia di lingkungan Anda disalahgunakan.

Anda dapat mengatur seluruh string dalam nilai rahasia sebagai nilai variabel lingkungan, atau jika string adalah JSON, Anda dapat mengurai JSON untuk mengatur variabel lingkungan individu untuk setiap pasangan nilai kunci JSON. Jika nilai rahasia adalah biner, tindakan mengubahnya menjadi string.

Untuk melihat variabel lingkungan yang dibuat dari rahasia Anda, aktifkan logging debug. Untuk informasi selengkapnya, lihat Mengaktifkan logging debug di Dokumen. GitHub

Untuk menggunakan variabel lingkungan yang dibuat dari rahasia Anda, lihat Variabel lingkungan di GitHub Dokumen.

Prasyarat

Untuk menggunakan tindakan ini, Anda harus terlebih dahulu mengonfigurasi AWS kredensil dan mengatur Wilayah AWS di GitHub lingkungan Anda dengan menggunakan langkah tersebut`configure-aws-credentials`. Ikuti petunjuk di Mengonfigurasi Tindakan AWS Kredensial Untuk GitHub Tindakan untuk Mengasumsikan peran secara langsung menggunakan penyedia GitHub OIDC. Ini memungkinkan Anda untuk menggunakan kredensil berumur pendek dan menghindari menyimpan kunci akses tambahan di luar Secrets Manager.

Peran IAM yang diasumsikan tindakan harus memiliki izin berikut:

- `GetSecretValue`pada rahasia yang ingin Anda ambil.
- `ListSecrets`pada semua rahasia.
- (Opsional) `Decrypt` pada KMS key jika rahasia dienkripsi dengan file. kunci yang dikelola pelanggan

Untuk informasi selengkapnya, lihat the section called “Kontrol autentikasi dan akses”.

Penggunaan

Untuk menggunakan tindakan, tambahkan langkah ke alur kerja Anda yang menggunakan sintaks berikut.

```
- name: Step name
  uses: aws-actions/aws-secretsmanager-get-secrets@v2
  with:
    secret-ids: |
      secretId1
      ENV_VAR_NAME, secretId2
    name-transformation: (Optional) uppercase|lowercase|none
    parse-json-secrets: (Optional) true|false
```

Parameter

`secret-ids`

Rahasia ARNS, nama, dan awalan nama.

Untuk mengatur nama variabel lingkungan, masukkan sebelum ID rahasia, diikuti dengan koma. Misalnya `ENV_VAR_1, secretId` membuat variabel lingkungan bernama ENV_VAR_1 dari rahasia. `secretId` Nama variabel lingkungan dapat terdiri dari huruf besar, angka, dan garis bawah.

Untuk menggunakan awalan, masukkan setidaknya tiga karakter diikuti dengan tanda bintang. Misalnya `dev*` mencocokkan semua rahasia dengan nama yang dimulai di dev. Jumlah maksimum rahasia pencocokan yang dapat diambil adalah 100. Jika Anda menetapkan nama variabel, dan awalan cocok dengan beberapa rahasia, maka tindakan gagal.

`name-transformation`

Secara default, langkah membuat setiap nama variabel lingkungan dari nama rahasia, diubah untuk menyertakan hanya huruf besar, angka, dan garis bawah, dan agar tidak dimulai dengan angka. Untuk huruf dalam nama, Anda dapat mengonfigurasi langkah untuk menggunakan huruf kecil dengan `lowercase` atau tidak mengubah huruf dengan. `none` Nilai default-nya adalah `uppercase`.

`parse-json-secrets`

(Opsional) Secara default, tindakan menetapkan nilai variabel lingkungan ke seluruh string JSON dalam nilai rahasia. Atur `parse-json-secrets` `true` untuk membuat variabel lingkungan untuk setiap pasangan kunci-nilai di JSON.

Perhatikan bahwa jika JSON menggunakan kunci peka huruf besar/kecil seperti “nama” dan “Nama”, tindakan akan memiliki konflik nama duplikat. Dalam hal ini, atur `parse-json-secrets` ke `false` dan parse nilai rahasia JSON secara terpisah.

Penamaan variabel lingkungan

Variabel lingkungan yang dibuat oleh tindakan diberi nama sama dengan rahasia tempat mereka berasal. Variabel lingkungan memiliki persyaratan penamaan yang lebih ketat daripada rahasia, sehingga tindakan mengubah nama rahasia untuk memenuhi persyaratan tersebut. Misalnya, tindakan mengubah huruf kecil menjadi huruf besar. Jika Anda mengurai JSON rahasia, maka nama variabel lingkungan mencakup nama rahasia dan nama kunci JSON, misalnya. `MYSECRET_KEYNAME` Anda dapat mengonfigurasi tindakan untuk tidak mengubah huruf kecil.

Jika dua variabel lingkungan akan berakhir dengan nama yang sama, tindakan gagal. Dalam hal ini, Anda harus menentukan nama yang ingin Anda gunakan untuk variabel lingkungan sebagai alias.

Contoh kapan nama mungkin bertentangan:

- Sebuah rahasia bernama "MySecret" dan rahasia bernama “mysecret” keduanya akan menjadi variabel lingkungan bernama “MYSECRET”.
- Sebuah rahasia bernama "Secret_keyname" dan JSON-parsed rahasia bernama “Rahasia” dengan kunci bernama “keyname” keduanya akan menjadi variabel lingkungan bernama “SECRET_KEYNAME”.

Anda dapat mengatur nama variabel lingkungan dengan menentukan alias, seperti yang ditunjukkan pada contoh berikut, yang menciptakan variabel bernama. `ENV_VAR_NAME`

```
secret-ids: |
  ENV_VAR_NAME, secretId2
```

Alias kosong

- Jika Anda mengatur `parse-json-secrets: true` dan memasukkan alias kosong, diikuti dengan koma dan kemudian ID rahasia, tindakan tersebut memberi nama variabel lingkungan sama dengan kunci JSON yang diurai. Nama variabel tidak termasuk nama rahasia.

  Jika rahasia tidak berisi JSON yang valid, maka tindakan akan membuat satu variabel lingkungan dan menamainya sama dengan nama rahasia.
- Jika Anda mengatur `parse-json-secrets: false` dan memasukkan alias kosong, diikuti dengan koma dan ID rahasia, tindakan tersebut memberi nama variabel lingkungan seolah-olah Anda tidak menentukan alias.

Contoh berikut menunjukkan alias kosong.

```
,secret2
```

Contoh

Example 1 Dapatkan rahasia dengan nama dan oleh ARN

Contoh berikut menciptakan variabel lingkungan untuk rahasia diidentifikasi dengan nama dan oleh ARN.

```
- name: Get secrets by name and by ARN
```

```
  uses: aws-actions/aws-secretsmanager-get-secrets@v2
  with:
    secret-ids: |
      exampleSecretName
      arn:aws:secretsmanager:us-east-2:123456789012:secret:test1-a1b2c3
      0/test/secret
      /prod/example/secret
      SECRET_ALIAS_1,test/secret
      SECRET_ALIAS_2,arn:aws:secretsmanager:us-east-2:123456789012:secret:test2-a1b2c3
      ,secret2
```

Variabel lingkungan dibuat:

```
EXAMPLESECRETNAME: secretValue1
TEST1: secretValue2
_0_TEST_SECRET: secretValue3
_PROD_EXAMPLE_SECRET: secretValue4
SECRET_ALIAS_1: secretValue5
SECRET_ALIAS_2: secretValue6
SECRET2: secretValue7
```

Example 2 Dapatkan semua rahasia yang dimulai dengan awalan

Contoh berikut menciptakan variabel lingkungan untuk semua rahasia dengan nama yang dimulai dengan*beta*.

```
- name: Get Secret Names by Prefix
  uses: 2
  with:
    secret-ids: |
      beta*    # Retrieves all secrets that start with 'beta'
```

Variabel lingkungan dibuat:

```
BETASECRETNAME: secretValue1
BETATEST: secretValue2
BETA_NEWSECRET: secretValue3
```

Example 3 Parse JSON secara rahasia

Contoh berikut menciptakan variabel lingkungan dengan mengurai JSON dalam rahasia.

```
- name: Get Secrets by Name and by ARN
  uses: aws-actions/aws-secretsmanager-get-secrets@v2
  with:
    secret-ids: |
      test/secret
      ,secret2
    parse-json-secrets: true
```

Rahasianya `test/secret` memiliki nilai rahasia berikut.

```
{
  "api_user": "user",
  "api_key": "key",
  "config": {
    "active": "true"
  }
}
```

Rahasianya `secret2` memiliki nilai rahasia berikut.

```
{
  "myusername": "alejandro_rosalez",
  "mypassword": "EXAMPLE_PASSWORD"
}
```

Variabel lingkungan dibuat:

```
TEST_SECRET_API_USER: "user"
TEST_SECRET_API_KEY: "key"
TEST_SECRET_CONFIG_ACTIVE: "true"
MYUSERNAME: "alejandro_rosalez"
MYPASSWORD: "EXAMPLE_PASSWORD"
```

Example 4 Gunakan huruf kecil untuk nama variabel lingkungan

Contoh berikut menciptakan variabel lingkungan dengan nama huruf kecil.

```
- name: Get secrets
  uses: aws-actions/aws-secretsmanager-get-secrets@v2
  with:
    secret-ids: exampleSecretName
```

```
    name-transformation: lowercase
```

Variabel lingkungan dibuat:

```
examplesecretname: secretValue
```

Gunakan AWS Secrets Manager di GitLab

AWS Secrets Manager terintegrasi dengan GitLab. Anda dapat memanfaatkan rahasia Manajer Rahasia untuk melindungi GitLab kredensibilitas Anda sehingga tidak lagi di-hardcode. GitLab Sebagai gant GitLab inya, Runner mengambil rahasia ini dari Secrets Manager saat aplikasi Anda menjalankan pekerjaan di GitLab CI/CD pipeline.

Untuk menggunakan integrasi ini, Anda akan membuat penyedia identitas OpenID Connect (OIDC) di IAM AWS Identity and Access Management dan peran IAM. Ini memungkinkan GitLab Runner untuk mengakses rahasia Manajer Rahasia Anda. Untuk informasi selengkapnya tentang GitLab CI/CD dan OIDC, lihat GitLab dokumentasi.

Pertimbangan-pertimbangan

Jika Anda menggunakan GitLab instance non-publik, Anda tidak dapat menggunakan integrasi Manajer Rahasia ini. Sebagai gantinya, lihat GitLab dokumentasi untuk inst ans non-publik.

Prasyarat

Untuk mengintegrasikan Secrets Manager dengan GitLab, lengkapi prasyarat berikut:

1. Buat AWS Secrets Manager Rahasia

   Anda akan memerlukan rahasia Manajer Rahasia yang akan diambil di GitLab pekerjaan Anda dan menghilangkan kebutuhan untuk mengkodekan kredenSIAL ini. Anda akan memerlukan ID rahasia Manajer Rahasia saat mengonfigurasi GitLab pipeline Anda. Untuk informasi selengkapnya, lihat Buat AWS Secrets Manager Rahasia.
2. GitLab Buat penyedia OIDC Anda di konsol IAM.

   Pada langkah ini, Anda akan membuat penyedia OIDC GitLab Anda di konsol IAM. Untuk informasi selengkapnya, lihat Membuat penyedia identitas OpenID Connect (OIDC) dan GitLab dokumentasi.

Saat membuat penyedia OIDC di konsol IAM, gunakan konfigurasi berikut:

a. Atur `provider URL` ke GitLab instance Anda. Misalnya, **`gitlab.example.com`**.

b. Atur `audience` atau `aud` ke**`sts.amazonaws.com`**.

3. Membuat peran dan kebijakan IAM

Anda harus membuat peran dan kebijakan IAM. Peran ini diasumsikan oleh GitLab dengan AWS Security Token Service (STS). Lihat Membuat peran menggunakan kebijakan kepercayaan khusus untuk informasi selengkapnya.

a. Di konsol IAM, gunakan pengaturan berikut saat membuat peran IAM:

- Atur `Trusted entity type` ke **`Web identity`**.
- Atur `Group` ke **`your GitLab group`**.
- Setel `Identity provider` ke URL penyedia yang sama (GitLab instance) yang Anda gunakan pada langkah 2.
- Setel `Audience` ke audiens yang sama dengan yang Anda gunakan di langkah 2.

b. Berikut ini adalah contoh kebijakan kepercayaan yang memungkinkan GitLab untuk mengambil peran. Kebijakan kepercayaan Anda harus mencantumkan GitLab URL Akun AWS, dan jalur proyek Anda.

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:oidc-
provider/gitlab.example.com"
      },
      "Condition": {
        "StringEquals": {
          "gitlab.example.com:aud": [
            "sts.amazon.com"
          ]
        },
```

```
        "StringLike": {
          "gitlab.example.com:sub": [
            "project_path:mygroup/project-*:ref_type:branch-*:ref:main*"
          ]
        }
      }
    }
  ]
}
```

c. Anda juga harus membuat kebijakan IAM untuk mengizinkan GitLab akses ke AWS Secrets Manager. Anda dapat menambahkan kebijakan ini ke kebijakan kepercayaan Anda. Untuk informasi selengkapnya, lihat Membuat kebijakan IAM.

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "secretsmanager:GetSecretValue",
      "Resource": "arn:aws:secretsmanager:us-
east-1:111122223333:secret:your-secret"
    }
  ]
}
```

Mengintegrasikan AWS Secrets Manager dengan GitLab

Setelah menyelesaikan prasyarat, Anda dapat mengonfigurasi GitLab untuk menggunakan Manajer Rahasia untuk melindungi kredenSIAL Anda.

Konfigur GitLab asikan pipeline untuk menggunakan Secrets Manager

Anda harus memperbarui file GitLab CI/CD konfigurasi Anda dengan informasi berikut:

- Penonton token diatur ke STS.
- ID rahasia Manajer Rahasia.
- Peran IAM yang Anda ingin GitLab Runner ambil saat menjalankan pekerjaan dalam GitLab pipeline.

- Dim Wilayah AWS ana rahasianya disimpan.

GitLab mengambil rahasia dari Secrets Manager dan menyimpan nilainya dalam file sementara. Jalur ke file ini disimpan dalam CI/CD variabel, mirip dengan CI/CD variabel tipe file.

Berikut ini adalah cuplikan file YAML untuk file GitLab CI/CD konfigurasi:

```
variables:
  AWS_REGION: us-east-1
  AWS_ROLE_ARN: 'arn:aws:iam::111122223333:role/gitlab-role'
job:
  id_tokens:
    AWS_ID_TOKEN:
      aud: 'sts.amazonaws.com'
  secrets:
    DATABASE_PASSWORD:
      aws_secrets_manager:
        secret_id: "arn:aws:secretsmanager:us-east-1:111122223333:secret:secret-name"
```

Untuk informasi selengkapnya, lihat dokumentasi integrasi GitLab Secrets Manager.

Secara opsional, Anda dapat menguji konfigurasi OIDC Anda di GitLab. Lihat GitLab dokumentasi untuk menguji konfigurasi OIDC untuk informasi selengkapnya.

Pemecahan masalah

Berikut ini dapat membantu Anda memecahkan masalah umum yang mungkin Anda temui saat mengintegrasikan Secrets Manager dengan GitLab.

GitLab Masalah pipa

Jika Anda mengalami masalah GitLab pipeline, pastikan hal berikut:

- File YAML Anda diformat dengan benar. Untuk informasi selengkapnya, lihat GitLab dokumentasi.
- GitLab Pipeline Anda mengambil peran yang benar, memiliki izin yang sesuai, dan akses ke AWS Secrets Manager rahasia yang benar.

Sumber daya tambahan

Sumber daya berikut dapat membantu Anda memecahkan masalah dengan GitLab dan AWS Secrets Manager:

- GitLab Pemecahan masalah OIDC
- GitLab CI/CD Pipa Debugging
- Pemecahan masalah

Gunakan AWS Secrets Manager rahasia di AWS IoT Greengrass

AWS IoT Greengrass adalah perangkat lunak yang memperluas kemampuan cloud ke perangkat lokal. Hal ini memungkinkan perangkat untuk mengumpulkan dan menganalisis data lebih dekat ke sumber informasi, bereaksi secara mandiri terhadap acara lokal, dan berkomunikasi secara aman satu sama lain di jaringan lokal.

AWS IoT Greengrass memungkinkan Anda mengautentikasi dengan layanan dan aplikasi dari AWS IoT Greengrass perangkat tanpa kata sandi hard-coding, token, atau rahasia lainnya. Anda dapat menggunakannya AWS Secrets Manager untuk menyimpan dan mengelola rahasia Anda dengan aman di cloud. AWS IoT Greengrass memperluas Secrets Manager ke perangkat AWS IoT Greengrass inti, sehingga konektor dan fungsi Lambda Anda dapat menggunakan rahasia lokal untuk berinteraksi dengan layanan dan aplikasi.

Untuk mengintegrasikan rahasia ke dalam AWS IoT Greengrass grup, Anda membuat sumber daya grup yang mereferensikan rahasia Secrets Manager. Sumber daya rahasia ini mereferensikan rahasia cloud dengan menggunakan ARN terkait. Untuk mempelajari cara membuat, mengelola, dan menggunakan sumber daya rahasia, lihat Bekerja dengan Sumber Daya Rahasia di Panduan AWS IoT Pengembang.

Untuk menyebarkan rahasia ke AWS IoT Greengrass Core, lihat Menyebarkan rahasia ke inti. AWS IoT Greengrass

Gunakan AWS Secrets Manager rahasia di Parameter Store

AWS Systems Manager Parameter Store menyediakan penyimpanan hierarkis yang aman untuk manajemen data konfigurasi dan manajemen rahasia. Anda dapat menyimpan data seperti kata sandi, string database, dan kode lisensi sebagai nilai parameter. Namun, Parameter Store tidak

menyediakan layanan rotasi otomatis untuk rahasia yang disimpan. Sebagai gantinya, Parameter Store memungkinkan Anda untuk menyimpan rahasia Anda di Secrets Manager, dan kemudian mereferensikan rahasia sebagai parameter Parameter Store.

Saat Anda mengonfigurasi Parameter Store dengan Secrets Manager, `secret-id` Parameter Store memerlukan garis miring (/) sebelum nama-string.

Untuk informasi selengkapnya, lihat Mereferensikan AWS Secrets Manager Rahasia dari Parameter Penyimpanan Parameter di Panduan AWS Systems Manager Pengguna.

Putar AWS Secrets Manager rahasia

Rotasi adalah proses memperbarui rahasia secara berkala. Ketika Anda memutar rahasia, Anda memperbarui kredensi di kedua rahasia dan database atau layanan. Di Secrets Manager, Anda dapat mengatur rotasi otomatis untuk rahasia Anda. Ada dua bentuk rotasi:

- Rotasi terkelola— Untuk sebagian besar rahasia terkelola, Anda menggunakan rotasi terkelola, tempat layanan mengonfigurasi dan mengelola rotasi untuk Anda. Rotasi terkelola tidak menggunakan fungsi Lambda.
- Rotate Secrets Manager mengelola rahasia eksternal— Untuk rahasia yang dipegang oleh mitra Secrets Manager, Anda menggunakan rotasi rahasia eksternal terkelola untuk memperbarui rahasia pada sistem mitra. Ini tidak memerlukan fungsi Lambda.
- the section called “Rotasi dengan fungsi Lambda”— Untuk jenis rahasia lainnya, rotasi Secrets Manager menggunakan fungsi Lambda untuk memperbarui rahasia dan database atau layanan.

Rotasi terkelola untuk AWS Secrets Manager rahasia

Beberapa layanan menawarkan rotasi terkelola; rotasi akan dikonfigurasi dan dikelola oleh layanan tersebut untuk Anda. Dengan rotasi terkelola, Anda tidak menggunakan AWS Lambda fungsi untuk memperbarui rahasia dan kredenSIAL dalam database.

Layanan berikut menawarkan rotasi terkelola:

- Amazon Aurora menawarkan rotasi terkelola untuk kredenSIAL pengguna utama. Untuk informasi selengkapnya, lihat Manajemen kata sandi dengan Amazon Aurora dan AWS Secrets Manager di Panduan Pengguna Amazon Aurora.
- Amazon ECS Service Connect menawarkan rotasi terkelola untuk sertifikat AWS Private Certificate Authority TLS. Untuk informasi selengkapnya, lihat TLS with Service Connect di Panduan Pengembang Amazon Elastic Container Service.
- Amazon RDS menawarkan rotasi terkelola untuk kredenSIAL pengguna utama. Untuk informasi selengkapnya, lihat Manajemen kata sandi dengan Amazon RDS dan AWS Secrets Manager di Panduan Pengguna Amazon RDS.
- Amazon DocumentDB menawarkan rotasi terkelola untuk kredensional pengguna utama. Untuk informasi selengkapnya, lihat Manajemen kata sandi dengan Amazon DocumentDB dan AWS Secrets Manager di Panduan Pengguna Amazon DocumentDB.

- Amazon Redshift menawarkan rotasi terkelola untuk kata sandi admin. Untuk informasi selengkapnya, lihat Mengelola kata sandi admin Amazon Redshift menggunakan AWS Secrets Manager di Panduan Manajemen Amazon Redshift.
- rahasia eksternal yang dikelola menawarkan rotasi terkelola untuk rahasia yang dipegang oleh mitra Manajer Rahasia. Untuk informasi selengkapnya, lihat Penggunaan AWS Secrets Manager rahasia eksternal yang dikelola untuk mengelola rahasia Pihak Ketiga.

> **Tip**
>
> Untuk semua jenis rahasia lainnya, lihatthe section called “Rotasi dengan fungsi Lambda”.

Rotasi untuk rahasia terkelola biasanya selesai dalam satu menit. Selama rotasi, koneksi baru yang mengambil rahasia mungkin mendapatkan versi kredenSIAL sebelumnya. Dalam aplikasi, kami sangat menyarankan agar Anda mengikuti praktik terbaik menggunakan pengguna database yang dibuat dengan hak istimewa minimal yang diperlukan untuk aplikasi Anda, daripada menggunakan pengguna master. Untuk pengguna aplikasi, untuk ketersediaan tertinggi, Anda dapat menggunakan strategi rotasi pengguna alternatif.

Untuk rahasia yang dipegang oleh mitra Manajer Rahasia,

Untuk mengubah jadwal rotasi terkelola

1. Buka rahasia terkelola di konsol Secrets Manager. Anda dapat mengikuti tautan dari layanan pengelolaan, atau mencari rahasia di konsol Manajer Rahasia.
2. Di bawah Jadwal rotasi, masukkan jadwal Anda di zona waktu UTC baik di pembuat ekspresi Jadwal atau sebagai ekspresi Jadwal. Manajer Rahasia menyimpan jadwal Anda sebagai `cron()` ekspresi `rate()` atau. Jendela rotasi secara otomatis dimulai pada tengah malam kecuali Anda menentukan waktu mulai. Anda dapat merotasi rahasia setiap empat jam. Untuk informasi selengkapnya, lihat Jadwal rotasi.
3. (Opsional) Untuk durasi jendela, pilih panjang jendela di mana Anda ingin Manajer Rahasia memutar rahasia Anda, **3h** misalnya untuk jendela tiga jam. Jendela tidak boleh meluas ke jendela rotasi berikutnya. Jika Anda tidak menentukan Durasi jendela, untuk jadwal rotasi dalam jam, jendela akan ditutup secara otomatis setelah satu jam. Untuk jadwal rotasi dalam hari, jendela secara otomatis ditutup pada akhir hari.
4. Pilih Simpan.

Untuk mengubah jadwal rotasi terkelola (AWS CLI)

- Panggil `rotate-secret`. Contoh berikut memutar rahasia antara pukul 16:00 dan 18:00 UTC pada hari ke-1 dan ke-15 setiap bulan. Lihat informasi yang lebih lengkap di Jadwal rotasi.

```
aws secretsmanager rotate-secret \
    --secret-id MySecret \
    --rotation-rules \
        "{\"ScheduleExpression\": \"cron(0 16 1,15 * ? *)\", \"Duration\": \"2h\"}"
```

Rotate Secrets Manager mengelola rahasia eksternal

Secrets Manager telah bermitra dengan vendor perangkat lunak tertentu untuk menawarkan rahasia eksternal yang dikelola. Fitur ini membantu pelanggan mengelola siklus hidup rahasia dengan menangani rotasi secara otomatis. Dengan rahasia eksternal yang dikelola, pelanggan tidak perlu lagi mempertahankan logika rotasi spesifik untuk setiap rahasia yang disimpan dengan mitra yang berbeda. Ini akan ditangani oleh Secrets Manager.

Untuk melihat daftar mitra yang terhubung dengan Secrets Manager, lihat Partner rahasia eksternal yang dikelola.

Mengatur Rotasi di Konsol

Untuk mengonfigurasi rotasi rahasia eksternal terkelola yang ada, yang dibuat dengan menentukan jenis dan nilai rahasia seperti yang ditentukan oleh mitra integrasi masing-masing, gunakan langkah-langkah berikut:

1. Buka konsol Secrets Manager.
2. Pilih rahasia eksternal terkelola Anda dari daftar.
3. Pilih tab Konfigurasi.
4. Di bagian konfigurasi rotasi, pilih Edit rotasi.
5. Nyalakan rotasi otomatis.
6. Di bawah metadata Rotasi, tambahkan metadata khusus mitra apa pun yang diperlukan untuk rotasi:

   Ikuti panduan yang diberikan oleh mitra integrasi Anda untuk metadata lain yang diperlukan
7. Di Izin layanan untuk rotasi rahasia, pilih atau buat peran IAM untuk rotasi:

- Pilih Buat peran baru untuk membuat peran secara otomatis dengan izin yang diperlukan
- Atau pilih peran yang ada dengan izin yang sesuai untuk pasangan Anda

Secara default, izin dicakup oleh mitra individu di wilayah tempat rahasia dibuat

8. Atur jadwal Rotasi Anda (misalnya, putar secara otomatis setiap 30 hari).
9. Pilih Simpan untuk menerapkan konfigurasi rotasi.

Dua bidang metadata utama yang dikonfigurasi selama proses ini adalah:

Bidang	Deskripsi
ExternalSecretRotationMetadata	Partner-specific metadata yang diperlukan untuk rotasi, seperti versi API untuk Salesforce
ExternalSecretRotationRoleArn	ARN dari peran IAM yang digunakan untuk rotasi, dengan izin dicakup ke mitra integrasi

Untuk informasi selengkapnya tentang bidang ini, lihat Menggunakan rahasia eksternal yang dikelola Secrets Manager untuk mengelola rahasia Pihak Ketiga.

Mengatur Rotasi Menggunakan CLI

Jalankan perintah berikut untuk mengatur rotasi untuk rahasia Salesforce. Perintah ini menentukan ID rahasia, ARN peran IAM untuk rotasi, jadwal rotasi, dan metadata khusus mitra apa pun yang diperlukan untuk proses rotasi.

```
aws secretsmanager rotate-secret \
            --secret-id SampleSecret \
            --external-secret-rotation-role-arn arn:aws:iam::123412341234:role/xyz \
            --rotation-rules AutomaticallyAfterDays=1 \
            --external-secret-rotation-metadata
 '[{"Key":"apiVersion","Value":"v65.0"}]'
```

Rotasi dengan fungsi Lambda

Untuk banyak jenis rahasia, Secrets Manager menggunakan AWS Lambda fungsi untuk memperbarui rahasia dan database atau layanan. Untuk informasi tentang biaya penggunaan fungsi Lambda, lihat. Harga

Untuk beberapaRahasia dikelola oleh layanan lain, Anda menggunakan rotasi terkelola. Untuk menggunakanRotasi terkelola, Anda pertama kali membuat rahasia melalui layanan pengelolaan.

Selama rotasi, Secrets Manager mencatat peristiwa yang menunjukkan keadaan rotasi. Untuk informasi selengkapnya, lihat the section called “Masuk dengan AWS CloudTrail ”.

Untuk memutar rahasia, Secrets Manager memanggil fungsi Lambda sesuai dengan jadwal rotasi yang Anda atur. Jika Anda juga memperbarui nilai rahasia Anda secara manual saat rotasi otomatis diatur, maka Secrets Manager menganggap bahwa rotasi yang valid ketika menghitung tanggal rotasi berikutnya.

Selama rotasi, Secrets Manager memanggil fungsi yang sama beberapa kali, setiap kali dengan parameter yang berbeda. Secrets Manager memanggil fungsi dengan struktur permintaan parameter JSON berikut:

```
{
    "Step" : "request.type",
    "SecretId" : "string",
    "ClientRequestToken" : "string",
    "RotationToken" : "string"
}
```

Parameter:

- Langkah — Langkah rotasi:`create_secret`,`set_secret`,`test_secret`, atau`finish_secret`. Untuk informasi selengkapnya, lihat the section called “Empat langkah dalam fungsi rotasi”.
- SecretId— ARN rahasia untuk memutar.
- ClientRequestToken— Pengidentifikasi unik untuk versi baru rahasia. Nilai ini membantu memastikan idempotensi. Untuk informasi selengkapnya, lihat PutSecretValue: ClientRequestToken di Referensi AWS Secrets Manager API.
- RotationToken— Pengidentifikasi unik yang menunjukkan sumber permintaan. Diperlukan untuk rotasi rahasia menggunakan peran yang diasumsikan atau rotasi lintas akun, di mana Anda

memutar rahasia di satu akun dengan menggunakan fungsi rotasi Lambda di akun lain. Dalam kedua kasus, fungsi rotasi mengasumsikan peran IAM untuk memanggil Secrets Manager dan kemudian Secrets Manager menggunakan token rotasi untuk memvalidasi identitas peran IAM.

Kegagalan rotasi dapat terjadi karena masalah dengan fungsi Lambda Anda atau dengan Secrets Manager. Jika ada langkah rotasi yang gagal, Secrets Manager mencoba ulang seluruh proses rotasi beberapa kali selama jendela rotasi terbuka pada rahasia.

Topik

- Siapkan rotasi otomatis untuk rahasia Amazon RDS, Amazon Aurora, Amazon Redshift, atau Amazon DocumentDB
- Mengatur rotasi otomatis untuk rahasia non-database AWS Secrets Manager
- Siapkan rotasi otomatis menggunakan AWS CLI
- Strategi rotasi fungsi Lambda
- Fungsi rotasi Lambda
- AWS Secrets Manager templat fungsi rotasi
- Izin peran eksekusi fungsi rotasi Lambda untuk AWS Secrets Manager
- Akses jaringan untuk fungsi AWS Lambda rotasi
- Pemecahan Masalah AWS Secrets Manager rotasi

Siapkan rotasi otomatis untuk rahasia Amazon RDS, Amazon Aurora, Amazon Redshift, atau Amazon DocumentDB

Tutorial ini menjelaskan cara mengatur the section called “Rotasi dengan fungsi Lambda” rahasia database. Rotasi adalah proses memperbarui rahasia secara berkala. Ketika Anda memutar rahasia, Anda memperbarui kredensil di kedua rahasia dan database. Di Secrets Manager, Anda dapat mengatur rotasi otomatis untuk rahasia database Anda.

Untuk mengatur rotasi menggunakan konsol, Anda harus terlebih dahulu memilih strategi rotasi. Kemudian Anda mengonfigurasi rahasia rotasi, yang menciptakan fungsi rotasi Lambda jika Anda belum memilikinya. Konsol juga menetapkan izin untuk peran eksekusi fungsi Lambda. Langkah terakhir adalah memastikan bahwa fungsi rotasi Lambda dapat mengakses Secrets Manager dan database Anda melalui jaringan.

> **Warning**
>
> Untuk mengaktifkan rotasi otomatis, Anda harus memiliki izin untuk membuat peran eksekusi IAM untuk fungsi rotasi Lambda dan melampirkan kebijakan izin padanya. Anda membutuhkan keduanya `iam:CreateRole` dan `iam:AttachRolePolicy` izin. Pemberian izin ini memungkinkan identitas untuk memberikan izin apa pun kepada diri mereka sendiri.

Langkah:

- Langkah 1: Pilih strategi rotasi dan (opsional) buat rahasia superuser
- Langkah 2: Konfigurasikan rotasi dan buat fungsi rotasi
- Langkah 3: (Opsional) Tetapkan kondisi izin tambahan pada fungsi rotasi
- Langkah 4: Siapkan akses jaringan untuk fungsi rotasi
- Langkah selanjutnya

Langkah 1: Pilih strategi rotasi dan (opsional) buat rahasia superuser

Untuk informasi tentang strategi yang ditawarkan oleh Secrets Manager, lihatthe section called "Strategi rotasi fungsi Lambda".

Jika Anda memilih strategi pengguna bergantian, Anda harus Buat rahasia dan menyimpan kredenal superuser database di dalamnya. Anda memerlukan rahasia dengan kredensil superuser karena rotasi mengkloning pengguna pertama, dan sebagian besar pengguna tidak memiliki izin itu. Perhatikan bahwa Amazon RDS Proxy tidak mendukung strategi pengguna bergantian.

Langkah 2: Konfigurasikan rotasi dan buat fungsi rotasi

Untuk mengaktifkan rotasi untuk rahasia Amazon RDS, Amazon DocumentDB, atau Amazon Redshift

1. Buka konsol Secrets Manager di https://console.aws.amazon.com/secretsmanager/.
2. Pada halaman Rahasia, pilih rahasia Anda.
3. Pada halaman Detail rahasia, di bagian konfigurasi Rotasi, pilih Edit rotasi.
4. Dalam kotak dialog Edit konfigurasi rotasi, lakukan hal berikut:
    a. Nyalakan Rotasi otomatis.
    b. Di bawah Jadwal rotasi, masukkan jadwal Anda di zona waktu UTC baik di pembuat ekspresi Jadwal atau sebagai ekspresi Jadwal. Secrets Manager menyimpan jadwal Anda

sebagai `cron()` ekspresi `rate()` atau. Jendela rotasi secara otomatis dimulai pada tengah malam kecuali Anda menentukan waktu Mulai. Anda dapat merotasi rahasia setiap empat jam. Untuk informasi selengkapnya, lihat Jadwal rotasi.

c. (Opsional) Untuk durasi Jendela, pilih panjang jendela di mana Anda ingin Secrets Manager memutar rahasia Anda, **3h** misalnya untuk jendela tiga jam. Jendela tidak boleh meluas ke jendela rotasi berikutnya. Jika Anda tidak menentukan Durasi jendela, untuk jadwal rotasi dalam jam, jendela akan ditutup secara otomatis setelah satu jam. Untuk jadwal rotasi dalam beberapa hari, jendela secara otomatis ditutup pada akhir hari.

d. (Opsional) Pilih Putar segera ketika rahasia disimpan untuk memutar rahasia Anda ketika Anda menyimpan perubahan Anda. Jika Anda menghapus kotak centang, maka rotasi pertama akan dimulai pada jadwal yang Anda tetapkan.

   Jika rotasi gagal, misalnya karena Langkah 3 dan 4 belum selesai, Secrets Manager mencoba ulang proses rotasi beberapa kali.

e. Di bawah fungsi Rotasi, lakukan salah satu hal berikut:

   - Pilih Buat fungsi Lambda baru dan masukkan nama untuk fungsi baru Anda. Secrets Manager menambahkan `SecretsManager` ke awal nama fungsi. Secrets Manager membuat fungsi berdasarkan template yang sesuai dan menetapkan izin yang diperlukan untuk peran eksekusi Lambda.
   - Pilih Gunakan fungsi Lambda yang ada untuk menggunakan kembali fungsi rotasi yang Anda gunakan untuk rahasia lain. Fungsi rotasi yang tercantum di bawah Konfigurasi VPC yang direkomendasikan memiliki VPC dan grup keamanan yang sama dengan database, yang membantu fungsi mengakses database.

f. Untuk strategi Rotasi, pilih strategi Single user atau Alternating users. Untuk informasi selengkapnya, lihat the section called “Langkah 1: Pilih strategi rotasi dan (opsional) buat rahasia superuser”.

5. Pilih Simpan.

Langkah 3: (Opsional) Tetapkan kondisi izin tambahan pada fungsi rotasi

Dalam kebijakan sumber daya untuk fungsi rotasi Anda, sebaiknya sertakan kunci konteks `aws:SourceAccount`untuk membantu mencegah Lambda digunakan sebagai wakil yang bingung. Untuk beberapa AWS layanan, untuk menghindari skenario wakil yang membingungkan, AWS merekomendasikan agar Anda menggunakan kunci kondisi `aws:SourceAccount`global `aws:SourceArn`dan global. Namun, jika Anda menyertakan kondisi `aws:SourceArn` dalam

kebijakan fungsi rotasi Anda, fungsi rotasi hanya dapat digunakan untuk merotasi rahasia yang ditentukan oleh ARN tersebut. Sebaiknya hanya sertakan kunci konteks `aws:SourceAccount` agar Anda dapat menggunakan fungsi rotasi untuk beberapa rahasia.

Untuk memperbarui kebijakan sumber daya fungsi rotasi

1. Di konsol Secrets Manager, pilih rahasia Anda, dan kemudian pada halaman detail, di bawah konfigurasi Rotasi, pilih fungsi rotasi Lambda. Konsol Lambda terbuka.
2. Ikuti petunjuk di Menggunakan kebijakan berbasis sumber daya untuk Lambda untuk menambahkan kondisi. `aws:sourceAccount`

```
"Condition": {
    "StringEquals": {
        "AWS:SourceAccount": "123456789012"
    }
},
```

Jika rahasia dienkripsi dengan kunci KMS selain, Secrets Kunci yang dikelola AWS `aws/secretsmanager` Manager memberikan izin peran eksekusi Lambda untuk menggunakan kunci tersebut. Anda dapat menggunakan konteks enkripsi secretArn untuk membatasi penggunaan fungsi dekripsi, sehingga peran fungsi rotasi hanya memiliki akses untuk mendekripsi rahasia yang bertanggung jawab untuk berputar.

Untuk memperbarui peran eksekusi fungsi rotasi

1. Dari fungsi rotasi Lambda, pilih Konfigurasi, lalu di bawah Peran eksekusi, pilih nama Peran.
2. Ikuti petunjuk di Memodifikasi kebijakan izin peran untuk menambahkan kondisi. `kms:EncryptionContext:SecretARN`

```
"Condition": {
    "StringEquals": {
        "kms:EncryptionContext:SecretARN": "SecretARN"
    }
},
```

Langkah 4: Siapkan akses jaringan untuk fungsi rotasi

Untuk informasi selengkapnya, lihat the section called “Akses jaringan untuk fungsi AWS Lambda rotasi”.

Langkah selanjutnya

Lihat the section called “Memecahkan masalah rotasi”.

Mengatur rotasi otomatis untuk rahasia non-database AWS Secrets Manager

Tutorial ini menjelaskan cara mengatur the section called “Rotasi dengan fungsi Lambda” rahasia non-database. Rotasi adalah proses memperbarui rahasia secara berkala. Ketika Anda memutar rahasia, Anda memperbarui kredensi di kedua rahasia dan database atau layanan yang menjadi rahasia itu.

Untuk rahasia database, lihatRotasi otomatis untuk rahasia database (konsol).

> **Warning**
>
> Untuk mengaktifkan rotasi otomatis, Anda harus memiliki izin untuk membuat peran eksekusi IAM untuk fungsi rotasi Lambda dan melampirkan kebijakan izin padanya. Anda membutuhkan keduanya `iam:CreateRole` dan `iam:AttachRolePolicy` izin. Pemberian izin ini memungkinkan identitas untuk memberikan izin apa pun kepada diri mereka sendiri.

Langkah:

- Langkah 1: Buat fungsi rotasi generik
- Langkah 2: Tulis kode fungsi rotasi
- Langkah 3: Konfigurasikan rahasia untuk rotasi
- Langkah 4: Izinkan fungsi rotasi untuk mengakses Secrets Manager dan database atau layanan Anda
- Langkah 5: Izinkan Secrets Manager untuk menjalankan fungsi rotasi
- Langkah 6: Siapkan akses jaringan untuk fungsi rotasi
- Langkah selanjutnya

Langkah 1: Buat fungsi rotasi generik

Untuk memulai, buat fungsi rotasi Lambda. Ini tidak akan memiliki kode di dalamnya untuk memutar rahasia Anda, jadi Anda akan menulisnya di langkah selanjutnya. Untuk informasi tentang cara kerja fungsi rotasi, lihatthe section called “Fungsi rotasi Lambda”.

Di Wilayah yang didukung, Anda dapat menggunakan AWS Serverless Application Repository untuk membuat fungsi dari template. Untuk daftar Wilayah yang didukung, lihat AWS Serverless Application Repository FAQ. Di Wilayah lain, Anda membuat fungsi dari awal dan menyalin kode template ke dalam fungsi.

Untuk membuat fungsi rotasi generik

1. Untuk menentukan AWS Serverless Application Repository apakah didukung di Wilayah Anda, lihat AWS Serverless Application Repository titik akhir dan kuota di Referensi AWS Umum.
2. Lakukan salah satu tindakan berikut:
   - Jika AWS Serverless Application Repository didukung di Wilayah Anda:
     a. Di konsol Lambda, pilih Aplikasi dan kemudian pilih Buat aplikasi.
     b. Pada halaman Buat aplikasi, pilih tab Aplikasi Tanpa Server.
     c. Di kotak pencarian di bawah Aplikasi publik, masukkan**`SecretsManagerRotationTemplate`**.
     d. Pilih Tampilkan aplikasi yang membuat peran IAM kustom atau kebijakan sumber daya.
     e. Pilih SecretsManagerRotationTemplateubin.
     f. Pada halaman Tinjau, konfigurasikan, dan terapkan, di ubin Pengaturan aplikasi, isi bidang yang diperlukan.
        - Untuk titik akhir, masukkan titik akhir untuk Wilayah Anda, termasuk. **`https://`** Untuk daftar titik akhir, lihatthe section called “Titik akhir Manajer Rahasia”.
        - Untuk menempatkan fungsi Lambda dalam VPC, sertakan vpc dan vpc. SecurityGroupIds SubnetIds
     g. Pilih Deploy.
   - Jika AWS Serverless Application Repository tidak didukung di Wilayah Anda:
     a. Di konsol Lambda, pilih Functions dan kemudian pilih Create function.
     b. Di halaman Buat fungsi, lakukan langkah berikut:

i. Pilih Tulis dari awal.

ii. Untuk nama Fungsi, masukkan nama untuk fungsi rotasi Anda.

iii. Untuk Runtime, pilih Python 3.12.

iv. Pilih Buat fungsi.

Langkah 2: Tulis kode fungsi rotasi

Pada langkah ini, Anda menulis kode yang memperbarui rahasia dan layanan atau basis data yang menjadi tujuan rahasianya. Untuk informasi tentang fungsi rotasi, termasuk tips menulis fungsi rotasi Anda sendiri, lihatthe section called “Fungsi rotasi Lambda”. Anda juga dapat menggunakan Templat fungsi rotasi sebagai referensi.

Langkah 3: Konfigurasikan rahasia untuk rotasi

Pada langkah ini, Anda mengatur jadwal rotasi untuk rahasia Anda dan menghubungkan fungsi rotasi ke rahasia.

Untuk mengkonfigurasi rotasi dan membuat fungsi rotasi kosong

1. Buka konsol Secrets Manager di https://console.aws.amazon.com/secretsmanager/.
2. Pada halaman Rahasia, pilih rahasia Anda.
3. Pada halaman Detail rahasia, di bagian konfigurasi Rotasi, pilih Edit rotasi. Dalam kotak dialog Edit konfigurasi rotasi, lakukan hal berikut:

   a. Nyalakan Rotasi otomatis.

   b. Di bawah Jadwal rotasi, masukkan jadwal Anda di zona waktu UTC baik di pembuat ekspresi Jadwal atau sebagai ekspresi Jadwal. Secrets Manager menyimpan jadwal Anda sebagai `cron()` ekspresi `rate()` atau. Jendela rotasi secara otomatis dimulai pada tengah malam kecuali Anda menentukan waktu Mulai. Anda dapat merotasi rahasia setiap empat jam. Untuk informasi selengkapnya, lihat Jadwal rotasi.

   c. (Opsional) Untuk durasi Jendela, pilih panjang jendela di mana Anda ingin Secrets Manager memutar rahasia Anda, **3h** misalnya untuk jendela tiga jam. Jendela tidak boleh meluas ke jendela rotasi berikutnya. Jika Anda tidak menentukan Durasi jendela, untuk jadwal rotasi dalam jam, jendela akan ditutup secara otomatis setelah satu jam. Untuk jadwal rotasi dalam beberapa hari, jendela secara otomatis ditutup pada akhir hari.

d. (Opsional) Pilih Putar segera ketika rahasia disimpan untuk memutar rahasia Anda ketika Anda menyimpan perubahan Anda. Jika Anda menghapus kotak centang, maka rotasi pertama akan dimulai pada jadwal yang Anda tetapkan.

e. Di bawah fungsi Rotasi, pilih fungsi Lambda yang Anda buat di Langkah 1.

f. Pilih Simpan.

Langkah 4: Izinkan fungsi rotasi untuk mengakses Secrets Manager dan database atau layanan Anda

Fungsi rotasi Lambda memerlukan izin untuk mengakses rahasia di Secrets Manager, dan memerlukan izin untuk mengakses database atau layanan Anda. Pada langkah ini, Anda memberikan izin ini ke peran eksekusi Lambda. Jika rahasia dienkripsi dengan kunci KMS selain Kunci yang dikelola AWS `aws/secretsmanager`, maka Anda perlu memberikan izin peran eksekusi Lambda untuk menggunakan kunci tersebut. Anda dapat menggunakan konteks enkripsi secretArn untuk membatasi penggunaan fungsi dekripsi, sehingga peran fungsi rotasi hanya memiliki akses untuk mendekripsi rahasia yang bertanggung jawab untuk berputar. Untuk contoh kebijakan, lihatIzin untuk rotasi.

Untuk petunjuknya, lihat Peran eksekusi Lambda di Panduan AWS Lambda Pengembang.

Langkah 5: Izinkan Secrets Manager untuk menjalankan fungsi rotasi

Untuk mengizinkan Secrets Manager menjalankan fungsi rotasi pada jadwal rotasi yang Anda atur, Anda harus memberikan `lambda:InvokeFunction` izin kepada kepala layanan Secrets Manager dalam kebijakan sumber daya fungsi Lambda.

Dalam kebijakan sumber daya untuk fungsi rotasi Anda, sebaiknya sertakan kunci konteks `aws:SourceAccount`untuk membantu mencegah Lambda digunakan sebagai wakil yang bingung. Untuk beberapa AWS layanan, untuk menghindari skenario wakil yang membingungkan, AWS merekomendasikan agar Anda menggunakan kunci kondisi `aws:SourceAccount`global `aws:SourceArn`dan global. Namun, jika Anda menyertakan kondisi `aws:SourceArn` dalam kebijakan fungsi rotasi Anda, fungsi rotasi hanya dapat digunakan untuk merotasi rahasia yang ditentukan oleh ARN tersebut. Sebaiknya hanya sertakan kunci konteks `aws:SourceAccount` agar Anda dapat menggunakan fungsi rotasi untuk beberapa rahasia.

Untuk melampirkan kebijakan sumber daya ke fungsi Lambda, lihat Menggunakan kebijakan berbasis sumber daya untuk Lambda.

Kebijakan berikut memungkinkan Secrets Manager untuk menjalankan fungsi Lambda.

JSON

```
{
    "Version":"2012-10-17",
    "Id": "default",
    "Statement": [
    {
        "Effect": "Allow",
        "Principal": {
            "Service": "secretsmanager.amazonaws.com"
            },
        "Action": "lambda:InvokeFunction",
        "Condition": {
            "StringEquals": {
                "AWS:SourceAccount": "123456789012"
            }
        },
        "Resource": "arn:aws:lambda:us-east-1:123456789012:function:function-
name"
    }
    ]
}
```

Langkah 6: Siapkan akses jaringan untuk fungsi rotasi

Pada langkah ini, Anda mengizinkan fungsi rotasi untuk terhubung ke Secrets Manager dan layanan atau database rahasianya. Fungsi rotasi harus memiliki akses ke keduanya untuk dapat memutar rahasia. Lihat the section called “Akses jaringan untuk fungsi AWS Lambda rotasi”.

Langkah selanjutnya

Saat Anda mengonfigurasi rotasi di Langkah 3, Anda mengatur jadwal untuk memutar rahasia. Jika rotasi gagal saat dijadwalkan, Secrets Manager akan mencoba rotasi beberapa kali. Anda juga dapat memulai rotasi segera dengan mengikuti instruksi diPutar rahasia segera.

Jika rotasi gagal, lihatMemecahkan masalah rotasi.

Siapkan rotasi otomatis menggunakan AWS CLI

Tutorial ini menjelaskan cara mengatur the section called “Rotasi dengan fungsi Lambda” dengan menggunakan AWS CLI. Saat Anda memutar rahasia, Anda memperbarui kredenSIAL di rahasia dan database atau layanan yang digunakan rahasia tersebut.

Anda juga dapat mengatur rotasi menggunakan konsol. Untuk rahasia database, lihatRotasi otomatis untuk rahasia database (konsol). Untuk semua jenis rahasia lainnya, lihatRotasi otomatis untuk rahasia non-database (konsol).

Untuk mengatur rotasi menggunakan AWS CLI, jika Anda memutar rahasia database, Anda harus terlebih dahulu memilih strategi rotasi. Jika Anda memilih strategi pengguna bergantian, Anda harus menyimpan rahasia terpisah dengan kredenSIAL untuk superuser database. Selanjutnya, Anda menulis kode fungsi rotasi. Manajer Rahasia menyediakan templat yang dapat Anda gunakan untuk mendasarkan fungsi Anda. Kemudian Anda membuat fungsi Lambda dengan kode Anda dan mengatur izin untuk fungsi Lambda dan peran eksekusi Lambda. Langkah selanjutnya adalah memastikan bahwa fungsi Lambda dapat mengakses Manajer Rahasia dan database atau layanan Anda melalui jaringan. Akhirnya, Anda mengkonfigurasi rahasia untuk rotasi.

Langkah:

- Prasyarat untuk rahasia database: Pilih strategi rotasi
- Langkah 1: Tulis kode fungsi rotasi
- Langkah 2: Buat fungsi Lambda
- Langkah 3: Siapkan akses jaringan
- Langkah 4: Konfigurasikan rahasia untuk rotasi
- Langkah selanjutnya

Prasyarat untuk rahasia database: Pilih strategi rotasi

Untuk informasi tentang strategi yang ditawarkan oleh Manajer Rahasia, lihatthe section called “Strategi rotasi fungsi Lambda”.

Opsi 1: Strategi pengguna tunggal

Jika Anda memilih strategi pengguna tunggal, Anda dapat melanjutkan dengan Langkah 1.

Opsi 2: Strategi pengguna bergantian

Jika Anda memilih strategi pengguna bergantian, Anda harus:

- Buat rahasia dan simpan kredenSIAL superuser database di dalamnya. Anda memerlukan rahasia dengan kredenSIAL superuser karena rotasi pengguna bergantian mengkloning pengguna pertama, dan sebagian besar pengguna tidak memiliki izin itu.
- Tambahkan ARN rahasia superuser ke rahasia aslinya. Untuk informasi selengkapnya, lihat the section called “Struktur JSON dari sebuah rahasia”.

Perhatikan bahwa Amazon RDS Proxy tidak mendukung strategi pengguna bergantian.

Langkah 1: Tulis kode fungsi rotasi

Untuk memutar rahasia, Anda memerlukan fungsi rotasi. Fungsi rotasi adalah fungsi Lambda yang dipanggil Secrets Manager untuk memutar rahasia Anda. Untuk informasi selengkapnya, lihat the section called “Rotasi dengan fungsi Lambda”. Pada langkah ini, Anda menulis kode yang memperbarui rahasia dan layanan atau database yang digunakan untuk rahasianya.

Secrets Manager menyediakan template untuk rahasia database Amazon RDS, Amazon Aurora, Amazon Redshift, dan Amazon DocumentDB di. Templat fungsi rotasi

Untuk menulis kode fungsi rotasi

1. Lakukan salah satu tindakan berikut:
   - Periksa daftar template fungsi rotasi. Jika ada satu yang cocok dengan strategi layanan dan rotasi Anda, salin kodenya.
   - Untuk jenis rahasia lainnya, Anda menulis fungsi rotasi Anda sendiri. Untuk petunjuk, lihat the section called “Fungsi rotasi Lambda”.
2. Simpan file dalam file ZIP *`my-function.zip`* bersama dengan dependensi yang diperlukan.

Langkah 2: Buat fungsi Lambda

Pada langkah ini, Anda membuat fungsi Lambda menggunakan file ZIP yang Anda buat di Langkah 1. Anda juga mengatur peran eksekusi Lambda, yang merupakan peran yang diasumsikan Lambda saat fungsi dipanggil.

Untuk membuat fungsi rotasi Lambda dan peran eksekusi

1. Buat kebijakan kepercayaan untuk peran eksekusi Lambda dan simpan sebagai file JSON. Untuk contoh dan informasi lebih lanjut, lihatthe section called “Izin untuk rotasi”. Kebijakan tersebut harus:

- Izinkan peran untuk memanggil operasi Manajer Rahasia pada rahasia.
- Izinkan peran untuk memanggil layanan yang menjadi rahasia, misalnya, untuk membuat kata sandi baru.

2. Buat peran eksekusi Lambda dan terapkan kebijakan kepercayaan yang Anda buat pada langkah sebelumnya dengan memanggil `iam create-role`.

```
aws iam create-role \
    --role-name rotation-lambda-role \
    --assume-role-policy-document file://trust-policy.json
```

3. Buat fungsi Lambda dari file ZIP dengan memanggil `lambda create-function`.

```
aws lambda create-function \
  --function-name my-rotation-function \
  --runtime python3.12 \
  --zip-file fileb://my-function.zip \
  --handler lambda_function.lambda_handler \
  --role arn:aws:iam::123456789012:role/service-role/rotation-lambda-role
```

4. Tetapkan kebijakan sumber daya pada fungsi Lambda untuk mengizinkan Secrets Manager memanggilnya dengan memanggil. `lambda add-permission`

```
aws lambda add-permission \
  --function-name my-rotation-function \
  --action lambda:InvokeFunction \
  --statement-id SecretsManager \
  --principal secretsmanager.amazonaws.com \
  --source-account 123456789012
```

Langkah 3: Siapkan akses jaringan

Untuk informasi selengkapnya, lihat the section called “Akses jaringan untuk fungsi AWS Lambda rotasi”.

Langkah 4: Konfigurasikan rahasia untuk rotasi

Untuk mengaktifkan rotasi otomatis untuk rahasia Anda, hubungi `rotate-secret`. Anda dapat mengatur jadwal rotasi dengan ekspresi `rate()` jadwal `cron()` atau, dan Anda dapat mengatur durasi jendela rotasi. Untuk informasi selengkapnya, lihat the section called “Jadwal rotasi”.

```
aws secretsmanager rotate-secret \
    --secret-id MySecret \
    --rotation-lambda-arn arn:aws:lambda:aws-region:123456789012:function:my-rotation-
function \
    --rotation-rules "{\"ScheduleExpression\": \"cron(0 16 1,15 * ? *)\", \"Duration\":
 \"2h\"}"
```

Langkah selanjutnya

Lihat the section called “Memecahkan masalah rotasi”.

Strategi rotasi fungsi Lambda

Untukthe section called “Rotasi dengan fungsi Lambda”, untuk rahasia database, Secrets Manager menawarkan dua strategi rotasi.

Strategi rotasi: pengguna tunggal

Strategi ini memperbarui kredensi untuk satu pengguna dalam satu rahasia. Untuk instans Amazon RDS Db2, karena pengguna tidak dapat mengubah kata sandi mereka sendiri, Anda harus memberikan kredensi admin dalam rahasia terpisah. Ini adalah strategi rotasi paling sederhana, dan cocok untuk sebagian besar kasus penggunaan. Secara khusus, kami menyarankan Anda menggunakan strategi ini untuk kredensi untuk satu kali (ad hoc) atau pengguna interaktif.

Ketika rahasia berputar, koneksi database terbuka tidak terputus. Sementara rotasi sedang terjadi, ada periode waktu singkat antara ketika kata sandi dalam database berubah dan ketika rahasia diperbarui. Selama waktu ini, ada risiko rendah database menolak panggilan yang menggunakan kredenal yang diputar. Anda dapat mengurangi risiko ini dengan strategi coba lagi yang tepat. Setelah rotasi, koneksi baru menggunakan kredensi baru.

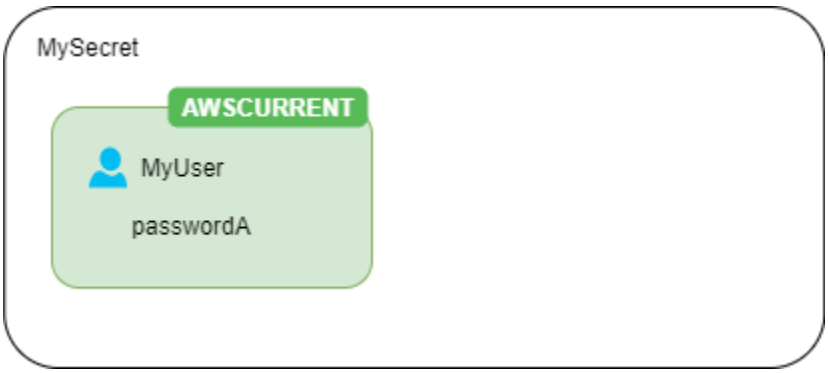
Strategi rotasi: pengguna bergantian

Strategi ini memperbarui kredensi untuk dua pengguna dalam satu rahasia. Anda membuat pengguna pertama, dan selama rotasi pertama, fungsi rotasi mengkloningnya untuk membuat pengguna kedua. Setiap kali rahasia dirotasi, fungsi rotasi secara bergantian memilih kata sandi pengguna mana yang akan diperbaruinya. Karena sebagian besar pengguna tidak memiliki izin untuk mengkloning diri mereka sendiri, Anda harus memberikan kredensialnya untuk rahasia lain. `superuser` Sebaiknya gunakan strategi rotasi pengguna tunggal ketika pengguna kloning di database Anda tidak memiliki izin yang sama dengan pengguna asli, dan untuk kredensi untuk pengguna satu kali (ad hoc) atau interaktif.

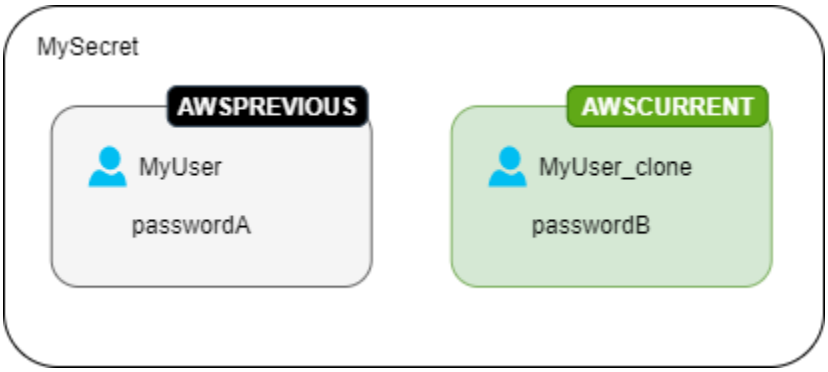
Strategi ini sesuai untuk database dengan model izin di mana satu peran memiliki tabel database dan peran kedua memiliki izin untuk mengakses tabel database. Ini juga sesuai untuk aplikasi yang membutuhkan ketersediaan tinggi. Jika aplikasi mengambil rahasia selama rotasi, aplikasi masih mendapatkan set kredensi yang valid. Setelah rotasi, keduanya `user` dan `user_clone` kredensialnya valid. Bahkan ada lebih sedikit kemungkinan aplikasi mendapatkan penolakan selama jenis rotasi ini daripada rotasi pengguna tunggal. Jika database di-host di server farm di mana perubahan kata sandi membutuhkan waktu untuk menyebar ke semua server, ada risiko database menolak panggilan yang menggunakan kredensi baru. Anda dapat mengurangi risiko ini dengan strategi coba lagi yang tepat.

Secrets Manager membuat pengguna kloning dengan izin yang sama dengan pengguna asli. Jika Anda mengubah izin pengguna asli setelah klon dibuat, Anda juga harus mengubah izin pengguna kloning.

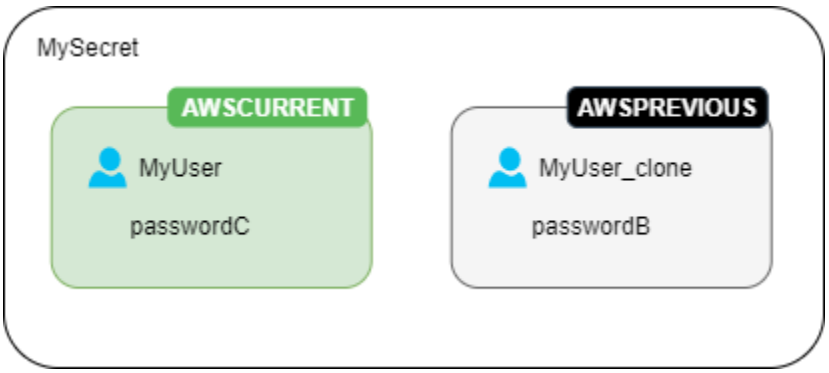
Misalnya, jika Anda membuat rahasia dengan kredensi pengguna database, rahasia berisi satu versi dengan kredensialnya.



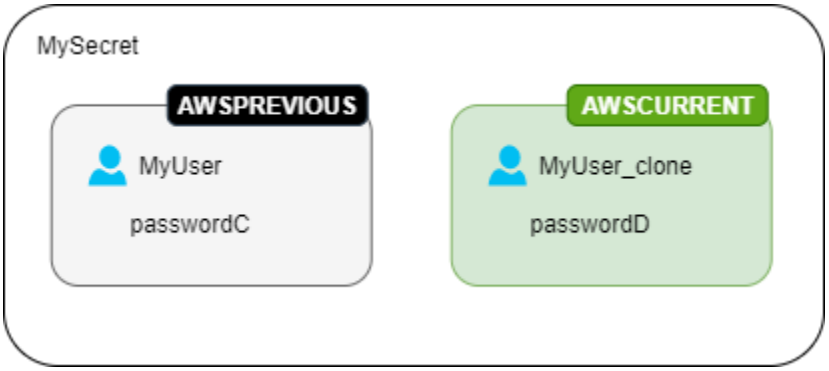
Rotasi pertama — Fungsi rotasi membuat tiruan pengguna Anda dengan kata sandi yang dihasilkan, dan kredensialnya menjadi versi rahasia saat ini.



Rotasi kedua — Fungsi rotasi memperbarui kata sandi untuk pengguna asli.



Rotasi ketiga - Fungsi rotasi memperbarui kata sandi untuk pengguna kloning.



Fungsi rotasi Lambda

Dalamthe section called “Rotasi dengan fungsi Lambda”, AWS Lambda fungsi memutar rahasia. AWS Secrets Manager menggunakan label pementasan untuk mengidentifikasi versi rahasia selama rotasi.

Jika AWS Secrets Manager tidak menyediakan template fungsi rotasi untuk tipe rahasia Anda, Anda dapat membuat fungsi rotasi kustom. Ikuti panduan ini saat menulis fungsi rotasi Anda:

Praktik terbaik untuk fungsi rotasi kustom

- Gunakan template rotasi generik sebagai titik awal.
- Berhati-hatilah dengan debugging atau logging statement. Mereka dapat menulis informasi ke Amazon CloudWatch Logs. Pastikan log tidak berisi informasi sensitif.

  Untuk contoh pernyataan log, lihat kode the section called “Templat fungsi rotasi” sumber.
- Untuk keamanan, AWS Secrets Manager hanya memungkinkan fungsi rotasi Lambda untuk memutar rahasia secara langsung. Fungsi rotasi tidak dapat memanggil fungsi Lambda lain untuk memutar rahasia.

- Untuk panduan debugging, lihat Menguji dan men-debug aplikasi tanpa server.
- Jika Anda menggunakan binari dan pustaka eksternal, misalnya untuk terhubung ke sumber daya, Anda bertanggung jawab untuk menambal dan memperbaruinya.
- Package fungsi rotasi Anda dan dependensi apa pun dalam file ZIP, seperti. *my-function.zip*

> **Warning**
>
> Menyetel parameter konkurensi yang disediakan ke nilai yang lebih rendah dari 10 dapat menyebabkan pelambatan karena utas eksekusi yang tidak mencukupi untuk fungsi Lambda. Untuk informasi selengkapnya, lihat Memahami konkurensi cadangan dan konkurensi yang disediakan di Panduan Pengembang. AWS Lambda AWS Lambda

Empat langkah dalam fungsi rotasi

Topik

- createSecret: Buat versi baru dari rahasia
- SETSECRET: Ubah kredensil dalam database atau layanan
- TESTSECRET: Uji versi rahasia baru
- FINISHSECRET: Selesaikan rotasi

`createSecret`: Buat versi baru dari rahasia

Metode `createSecret` pertama memeriksa apakah ada rahasia dengan memanggil `get_secret_value`dengan `ClientRequestToken` passed-in. Jika tidak ada rahasia, itu menciptakan rahasia baru dengan `create_secret`dan token sebagai`VersionId`. Kemudian menghasilkan nilai rahasia baru dengan `get_random_password`. Selanjutnya panggilan `put_secret_value`untuk menyimpannya dengan label `AWSPENDING` pementasan. Menyimpan nilai rahasia baru `AWSPENDING` membantu memastikan idempotensi. Jika rotasi gagal karena alasan apa pun, Anda dapat merujuk ke nilai rahasia itu dalam panggilan berikutnya. Lihat Bagaimana cara membuat fungsi Lambda saya idempoten.

Kiat untuk menulis fungsi rotasi Anda sendiri

- Pastikan nilai rahasia baru hanya mencakup karakter yang valid untuk database atau layanan. Kecualikan karakter dengan menggunakan `ExcludeCharacters` parameter.

- Saat Anda menguji fungsi Anda, gunakan AWS CLI untuk melihat tahapan versi: panggil `describe-secret`dan lihat`VersionIdsToStages`.
- Untuk Amazon RDS MySQL, dalam rotasi pengguna bergantian, Secrets Manager membuat pengguna kloning dengan nama tidak lebih dari 16 karakter. Anda dapat memodifikasi fungsi rotasi untuk memungkinkan nama pengguna yang lebih panjang. MySQL versi 5.7 dan yang lebih tinggi mendukung nama pengguna hingga 32 karakter, namun Secrets Manager menambahkan "_clone" (enam karakter) ke akhir nama pengguna, jadi Anda harus menjaga nama pengguna maksimal 26 karakter.

SETSECRET: Ubah kredensil dalam database atau layanan

Metode ini `setSecret` mengubah kredensi dalam database atau layanan untuk mencocokkan nilai rahasia baru dalam `AWSPENDING` versi rahasia.

Kiat untuk menulis fungsi rotasi Anda sendiri

- Jika Anda meneruskan pernyataan ke layanan yang menafsirkan pernyataan, seperti database, gunakan parameterisasi kueri. Untuk informasi selengkapnya, lihat Lembar Cheat Parameterisasi Kueri di situs web OWASP.
- Fungsi rotasi adalah wakil istimewa yang memiliki otorisasi untuk mengakses dan memodifikasi kredensil pelanggan baik dalam rahasia Secrets Manager dan sumber daya target. Untuk mencegah potensi serangan wakil yang membingungkan, Anda perlu memastikan bahwa penyerang tidak dapat menggunakan fungsi tersebut untuk mengakses sumber daya lain. Sebelum Anda memperbarui kredensialnya:
  - Periksa apakah kredensi dalam `AWSCURRENT` versi rahasia valid. Jika `AWSCURRENT` kredensialnya tidak valid, tinggalkan upaya rotasi.
  - Periksa apakah nilai `AWSCURRENT` dan `AWSPENDING` rahasia adalah untuk sumber daya yang sama. Untuk nama pengguna dan kata sandi, periksa apakah `AWSPENDING` nama pengguna `AWSCURRENT` dan nama pengguna sama.
  - Periksa apakah sumber daya layanan tujuan sama. Untuk database, periksa apakah nama `AWSCURRENT` dan `AWSPENDING` host sama.
- Dalam kasus yang jarang terjadi, Anda mungkin ingin menyesuaikan fungsi rotasi yang ada untuk database. Misalnya, dengan rotasi pengguna bergantian, Secrets Manager membuat pengguna kloning dengan menyalin parameter konfigurasi runtime dari pengguna pertama. Jika Anda ingin menyertakan lebih banyak atribut, atau mengubah mana yang diberikan kepada pengguna kloning, Anda perlu memperbarui kode dalam `set_secret` fungsi.

TESTSECRET: Uji versi rahasia baru

Selanjutnya, fungsi rotasi Lambda menguji `AWSPENDING` versi rahasia dengan menggunakannya untuk mengakses database atau layanan. Fungsi rotasi berdasarkan Templat fungsi rotasi uji rahasia baru dengan menggunakan akses baca.

FINISHSECRET: Selesaikan rotasi

Akhirnya, fungsi rotasi Lambda memindahkan label `AWSCURRENT` dari versi rahasia sebelumnya ke versi ini. Secrets Manager menambahkan label `AWSPREVIOUS` pementasan ke versi sebelumnya, sehingga Anda mempertahankan versi rahasia terakhir yang diketahui.

Metode ini finish_secret digunakan `update_secret_version_stage`untuk memindahkan label pementasan `AWSCURRENT` dari versi rahasia sebelumnya ke versi rahasia baru. Secrets Manager secara otomatis menambahkan label `AWSPREVIOUS` pementasan ke versi sebelumnya, sehingga Anda mempertahankan versi rahasia terakhir yang diketahui.

Kiat untuk menulis fungsi rotasi Anda sendiri

- Jangan hapus `AWSPENDING` sebelum titik ini, dan jangan hapus dengan menggunakan panggilan API terpisah, karena itu dapat menunjukkan kepada Secrets Manager bahwa rotasi tidak berhasil diselesaikan. Secrets Manager menambahkan label `AWSPREVIOUS` pementasan ke versi sebelumnya, sehingga Anda mempertahankan versi rahasia terakhir yang diketahui.

Ketika rotasi berhasil, label `AWSPENDING` pementasan mungkin dilampirkan ke versi yang sama dengan `AWSCURRENT` versi, atau mungkin tidak dilampirkan ke versi apa pun. Jika label `AWSPENDING` pementasan ada tetapi tidak dilampirkan ke versi yang sama dengan`AWSCURRENT`, maka pemanggilan rotasi selanjutnya mengasumsikan bahwa permintaan rotasi sebelumnya masih dalam proses dan mengembalikan kesalahan. Ketika rotasi tidak berhasil, label `AWSPENDING` pementasan mungkin dilampirkan ke versi rahasia kosong. Lihat informasi yang lebih lengkap di Memecahkan masalah rotasi.

AWS Secrets Manager templat fungsi rotasi

AWS Secrets Manager menyediakan satu set templat fungsi rotasi yang membantu mengotomatiskan manajemen kredensil yang aman untuk berbagai sistem dan layanan basis data. Template adalah fungsi Lambda siap pakai yang menerapkan praktik terbaik untuk rotasi kredensi, membantu Anda mempertahankan postur keamanan Anda tanpa intervensi manual.

Template mendukung dua strategi rotasi utama:

- Single-user rotasi yang memperbarui kredensil untuk satu pengguna.
- Alternating-users rotasi yang mempertahankan dua pengguna terpisah untuk membantu menghilangkan downtime selama perubahan kredensi.

Secrets Manager juga menyediakan template generik yang berfungsi sebagai titik awal untuk semua jenis rahasia.

Untuk menggunakan template, lihat:

- Rotasi otomatis untuk rahasia database (konsol)
- Rotasi otomatis untuk rahasia non-database (konsol)

Untuk menulis fungsi rotasi Anda sendiri, lihat Menulis fungsi rotasi.

Template

- Amazon RDS dan Amazon Aurora
  - Amazon RDS Db2 pengguna tunggal
  - Amazon RDS Db2 bergantian pengguna
  - Amazon RDS MariaDB pengguna tunggal
  - Amazon RDS MariaDB pengguna bergantian
  - Amazon RDS dan Amazon Aurora MySQL pengguna tunggal
  - Amazon RDS dan Amazon Aurora MySQL bergantian pengguna
  - Amazon RDS Oracle pengguna tunggal
  - Amazon RDS Oracle bergantian pengguna
  - Amazon RDS dan Amazon Aurora PostgreSQL pengguna tunggal
  - Amazon RDS dan Amazon Aurora PostgreSQL pengguna bergantian
  - Amazon RDS Microsoft SQLServer pengguna tunggal
  - Amazon RDS Microsoft SQLServer bergantian pengguna
- Amazon DocumentDB (dengan kompatibilitas MongoDB)
  - Amazon DocumentDB pengguna tunggal
  - Amazon DocumentDB pengguna bergantian
- Amazon Redshift
  - Amazon Redshift pengguna tunggal

  - Amazon Redshift bergantian pengguna
- Amazon Timestream untuk InfluxDB
  - Amazon Timestream untuk pengguna tunggal InfluxDB
  - Amazon Timestream untuk pengguna bergantian InfluxDB
- Amazon ElastiCache
- Active Directory
  - Kredensil Direktori Aktif
  - Tab tombol Direktori Aktif
- Jenis rahasia lainnya

Amazon RDS dan Amazon Aurora

Amazon RDS Db2 pengguna tunggal

- Nama template: SecretsManagerRDSDb2RotationSingleUser
- Strategi rotasi:Strategi rotasi: pengguna tunggal.
- **`SecretString`**struktur:the section called “Kredensi Amazon RDS dan Aurora”.
- Kode sumber: https://github.com/aws-samples/aws-secrets-manager-rotation-lambdas/tree/master/SecretsManagerRDSDb2RotationSingleUser/lambda_function.py
- Ketergantungan: python-ibmdb 3.2.8

Amazon RDS Db2 bergantian pengguna

- Nama template: SecretsManagerRDSDb2RotationMultiUser
- Strategi rotasi:the section called “Pengguna bergantian”.
- **`SecretString`**struktur:the section called “Kredensi Amazon RDS dan Aurora”.
- Kode sumber: https://github.com/aws-samples/aws-secrets-manager-rotation-lambdas/tree/master/SecretsManagerRDSDb2RotationMultiUser/lambda_function.py
- Ketergantungan: python-ibmdb 3.2.8

Amazon RDS MariaDB pengguna tunggal

- Nama template: SecretsManagerRDSMariaDBRotationSingleUser

- Strategi rotasi:Strategi rotasi: pengguna tunggal.
- **`SecretString`**struktur:the section called “Kredensi Amazon RDS dan Aurora”.
- Kode sumber: https://github.com/aws-samples/aws-secrets-manager-rotation-lambdas/tree/master/SecretsManagerRDSMariaDBRotationSingleUser/lambda_function.py
- Ketergantungan: PyMy SQL 1.1.1. Jika Anda menggunakan kata sandi sha256 untuk otentikasi, PyMy SQL [rsa]. Untuk informasi tentang penggunaan paket dengan kode yang dikompilasi dalam runtime Lambda, lihat Bagaimana cara menambahkan paket Python dengan binari yang dikompilasi ke paket penerapan saya dan membuat paket tersebut kompatibel dengan Lambda? di Pusat AWS Pengetahuan.

Amazon RDS MariaDB pengguna bergantian

- Nama template: SecretsManagerRDSMariaDBRotationMultiUser
- Strategi rotasi:the section called “Pengguna bergantian”.
- **`SecretString`**struktur:the section called “Kredensi Amazon RDS dan Aurora”.
- Kode sumber: https://github.com/aws-samples/aws-secrets-manager-rotation-lambdas/tree/master/SecretsManagerRDSMariaDBRotationMultiUser/lambda_function.py
- Ketergantungan: PyMy SQL 1.1.1. Jika Anda menggunakan kata sandi sha256 untuk otentikasi, PyMy SQL [rsa]. Untuk informasi tentang penggunaan paket dengan kode yang dikompilasi dalam runtime Lambda, lihat Bagaimana cara menambahkan paket Python dengan binari yang dikompilasi ke paket penerapan saya dan membuat paket tersebut kompatibel dengan Lambda? di Pusat AWS Pengetahuan.

Amazon RDS dan Amazon Aurora MySQL pengguna tunggal

- Nama template: SecretsManagerRDSMySQLRotationSingleUser
- Strategi rotasi:the section called “Pengguna tunggal”.
- **`SecretString`**Struktur yang diharapkan:the section called “Kredensi Amazon RDS dan Aurora”.
- Kode sumber: https://github.com/aws-samples/aws-secrets-manager-rotation-lambdas/tree/master/SecretsManagerRDSMySQLRotationSingleUser/lambda_function.py
- Ketergantungan: PyMy SQL 1.1.1. Jika Anda menggunakan kata sandi sha256 untuk otentikasi, PyMy SQL [rsa]. Untuk informasi tentang penggunaan paket dengan kode yang dikompilasi dalam runtime Lambda, lihat Bagaimana cara menambahkan paket Python dengan binari yang

dikompilasi ke paket penerapan saya dan membuat paket tersebut kompatibel dengan Lambda? di Pusat AWS Pengetahuan.

Amazon RDS dan Amazon Aurora MySQL bergantian pengguna

- Nama template: SecretsManagerRDSMySQLRotationMultiUser
- Strategi rotasi:the section called “Pengguna bergantian”.
- **`SecretString`**Struktur yang diharapkan:the section called “Kredensi Amazon RDS dan Aurora”.
- Kode sumber: https://github.com/aws-samples/aws-secrets-manager-rotation-lambdas/tree/master/SecretsManagerRDSMySQLRotationMultiUser/lambda_function.py
- Ketergantungan: PyMy SQL 1.1.1. Jika Anda menggunakan kata sandi sha256 untuk otentikasi, PyMy SQL [rsa]. Untuk informasi tentang penggunaan paket dengan kode yang dikompilasi dalam runtime Lambda, lihat Bagaimana cara menambahkan paket Python dengan binari yang dikompilasi ke paket penerapan saya dan membuat paket tersebut kompatibel dengan Lambda? di Pusat AWS Pengetahuan.

Amazon RDS Oracle pengguna tunggal

- Nama template: SecretsManagerRDSOracleRotationSingleUser
- Strategi rotasi:the section called “Pengguna tunggal”.
- **`SecretString`**Struktur yang diharapkan:the section called “Kredensi Amazon RDS dan Aurora”.
- Kode sumber: https://github.com/aws-samples/aws-secrets-manager-rotation-lambdas/tree/master/SecretsManagerRDSOracleRotationSingleUser/lambda_function.py
- Ketergantungan: python-oracledb 2.4.1

Amazon RDS Oracle bergantian pengguna

- Nama template: SecretsManagerRDSOracleRotationMultiUser
- Strategi rotasi:the section called “Pengguna bergantian”.
- **`SecretString`**Struktur yang diharapkan:the section called “Kredensi Amazon RDS dan Aurora”.
- Kode sumber: https://github.com/aws-samples/aws-secrets-manager-rotation-lambdas/tree/master/SecretsManagerRDSOracleRotationMultiUser/lambda_function.py
- Ketergantungan: python-oracledb 2.4.1

Amazon RDS dan Amazon Aurora PostgreSQL pengguna tunggal

- Nama template: SecretsManagerRDSPostgreSQLRotationSingleUser
- Strategi rotasi:Strategi rotasi: pengguna tunggal.
- **`SecretString`**Struktur yang diharapkan:the section called “Kredensi Amazon RDS dan Aurora”.
- Kode sumber: https://github.com/aws-samples/aws-secrets-manager-rotation-lambdas/tree/master/SecretsManagerRDSPostgreSQLRotationSingleUser/lambda_function.py
- Ketergantungan: PyGreSQL 6.1.0

Amazon RDS dan Amazon Aurora PostgreSQL pengguna bergantian

- Nama template: SecretsManagerRDSPostgreSQLRotationMultiUser
- Strategi rotasi:the section called “Pengguna bergantian”.
- **`SecretString`**Struktur yang diharapkan:the section called “Kredensi Amazon RDS dan Aurora”.
- Kode sumber: https://github.com/aws-samples/aws-secrets-manager-rotation-lambdas/tree/master/SecretsManagerRDSPostgreSQLRotationMultiUser/lambda_function.py
- Ketergantungan: PyGreSQL 6.1.0

Amazon RDS Microsoft SQLServer pengguna tunggal

- Nama template: SecretsManagerRDSSQLServerRotationSingleUser
- Strategi rotasi:the section called “Pengguna tunggal”.
- **`SecretString`**Struktur yang diharapkan:the section called “Kredensi Amazon RDS dan Aurora”.
- Kode sumber: https://github.com/aws-samples/aws-secrets-manager-rotation-lambdas/tree/master/SecretsManagerRDSSQLServerRotationSingleUser/lambda_function.py
- Ketergantungan: Pymssql 2.3.11

Amazon RDS Microsoft SQLServer bergantian pengguna

- Nama template: SecretsManagerRDSSQLServerRotationMultiUser
- Strategi rotasi:the section called “Pengguna bergantian”.
- **`SecretString`**Struktur yang diharapkan:the section called “Kredensi Amazon RDS dan Aurora”.
- Kode sumber: https://github.com/aws-samples/aws-secrets-manager-rotation-lambdas/tree/master/SecretsManagerRDSSQLServerRotationMultiUser/lambda_function.py

- Ketergantungan: Pymssql 2.3.11

Amazon DocumentDB (dengan kompatibilitas MongoDB)

Amazon DocumentDB pengguna tunggal

- Nama template: SecretsManagerMongoDBRotationSingleUser
- Strategi rotasi:the section called “Pengguna tunggal”.
- **`SecretString`**Struktur yang diharapkan:the section called “Kredensi Amazon DocumentDB”.
- Kode sumber: https://github.com/aws-samples/aws-secrets-manager-rotation-lambdas/tree/master/SecretsManagerMongoDBRotationSingleUser/lambda_function.py
- Ketergantungan: 4.7.3 PyMongo

Amazon DocumentDB pengguna bergantian

- Nama template: SecretsManagerMongoDBRotationMultiUser
- Strategi rotasi:the section called “Pengguna bergantian”.
- **`SecretString`**Struktur yang diharapkan:the section called “Kredensi Amazon DocumentDB”.
- Kode sumber: https://github.com/aws-samples/aws-secrets-manager-rotation-lambdas/tree/master/SecretsManagerMongoDBRotationMultiUser/lambda_function.py
- Ketergantungan: 4.7.3 PyMongo

Amazon Redshift

Amazon Redshift pengguna tunggal

- Nama template: SecretsManagerRedshiftRotationSingleUser
- Strategi rotasi:the section called “Pengguna tunggal”.
- **`SecretString`**Struktur yang diharapkan:the section called “Kredensi Amazon Redshift”.
- Kode sumber: https://github.com/aws-samples/aws-secrets-manager-rotation-lambdas/tree/master/SecretsManagerRedshiftRotationSingleUser/lambda_function.py
- Ketergantungan: konektor redshift-2.1.13

Amazon Redshift bergantian pengguna

- Nama template: SecretsManagerRedshiftRotationMultiUser
- Strategi rotasi:the section called “Pengguna bergantian”.
- **`SecretString`**Struktur yang diharapkan:the section called “Kredensi Amazon Redshift”.
- Kode sumber: https://github.com/aws-samples/aws-secrets-manager-rotation-lambdas/tree/master/SecretsManagerRedshiftRotationMultiUser/lambda_function.py
- Ketergantungan: konektor redshift-2.1.13

Amazon Timestream untuk InfluxDB

Untuk menggunakan templat ini, lihat Cara Amazon TimeStream untuk InfluxDB menggunakan rahasia di Panduan Pengembang Amazon Timestream.

Amazon Timestream untuk pengguna tunggal InfluxDB

- Nama template: SecretsManager InfluxDBRotationSingleUser
- **`SecretString`**Struktur yang diharapkan:the section called “Amazon Timestream untuk struktur rahasia InfluxDB”.
- Kode sumber: https://github.com/aws-samples/aws-secrets-manager-rotation-lambdas/tree/master/SecretsManagerInfluxDBRotationSingleUser/lambda_function.py
- Ketergantungan: Klien python InfluxDB 2.0

Amazon Timestream untuk pengguna bergantian InfluxDB

- Nama template: SecretsManagerInfluxDBRotationMultiUser
- **`SecretString`**Struktur yang diharapkan:the section called “Amazon Timestream untuk struktur rahasia InfluxDB”.
- Kode sumber: https://github.com/aws-samples/aws-secrets-manager-rotation-lambdas/tree/master/SecretsManagerInfluxDBRotationMultiUser/lambda_function.py
- Ketergantungan: Klien python InfluxDB 2.0

Amazon ElastiCache

Untuk menggunakan templat ini, lihat Memutar kata sandi secara otomatis untuk pengguna di Panduan ElastiCache Pengguna Amazon.

- Nama template: SecretsManagerElasticacheUserRotation
- **`SecretString`**Struktur yang diharapkan:the section called “ElastiCache Kredensi Amazon”.
- Kode sumber: https://github.com/aws-samples/aws-secrets-manager-rotation-lambdas/tree/master/SecretsManagerElasticacheUserRotation/lambda_function.py

Active Directory

Kredensil Direktori Aktif

- Nama template: SecretsManagerActiveDirectoryRotationSingleUser
- **`SecretString`**Struktur yang diharapkan:the section called “Kredensil Direktori Aktif”.
- Kode sumber: https://github.com/aws-samples/aws-secrets-manager-rotation-lambdas/tree/master/SecretsManagerActiveDirectoryRotationSingleUser/lambda_function.py

Tab tombol Direktori Aktif

- Nama template: SecretsManagerActiveDirectoryAndKeytabRotationSingleUser
- **`SecretString`**Struktur yang diharapkan:the section called “Kredensil Direktori Aktif”.
- Kode sumber: https://github.com/aws-samples/aws-secrets-manager-rotation-lambdas/tree/master/SecretsManagerActiveDirectoryAndKeytabRotationSingleUser/lambda_function.py
- Dependensi: msktutil

Jenis rahasia lainnya

Secrets Manager menyediakan template ini sebagai titik awal bagi Anda untuk membuat fungsi rotasi untuk semua jenis rahasia.

- Nama template: SecretsManagerRotationTemplate
- Kode sumber: https://github.com/aws-samples/aws-secrets-manager-rotation-lambdas/tree/master/SecretsManagerRotationTemplate/lambda_function.py

Izin peran eksekusi fungsi rotasi Lambda untuk AWS Secrets Manager

Karenathe section called “Rotasi dengan fungsi Lambda”, ketika Secrets Manager menggunakan fungsi Lambda untuk memutar rahasia, Lambda mengasumsikan peran eksekusi IAM dan

memberikan kredensil tersebut ke kode fungsi Lambda. Untuk petunjuk tentang cara mengatur rotasi otomatis, lihat:

- Rotasi otomatis untuk rahasia database (konsol)
- Rotasi otomatis untuk rahasia non-database (konsol)
- Rotasi otomatis (AWS CLI)

Contoh berikut menunjukkan kebijakan inline untuk peran eksekusi fungsi rotasi Lambda. Untuk membuat peran eksekusi dan melampirkan kebijakan izin, lihat peran AWS Lambda eksekusi.

Contoh:

- Kebijakan untuk peran eksekusi fungsi rotasi Lambda
- Pernyataan kebijakan untuk kunci yang dikelola pelanggan
- Pernyataan kebijakan untuk strategi pengguna bergantian

Kebijakan untuk peran eksekusi fungsi rotasi Lambda

Contoh kebijakan berikut memungkinkan fungsi rotasi untuk:

- Jalankan operasi Secrets Manager untuk*SecretARN*.
- Buat kata sandi baru.
- Siapkan konfigurasi yang diperlukan jika database atau layanan Anda berjalan di VPC. Lihat Mengonfigurasi fungsi Lambda untuk mengakses sumber daya di VPC.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "secretsmanager:UpdateSecretVersionStage"
            ],
```

```
            "Resource": "arn:aws:secretsmanager:us-
east-1:123456789012:secret:secretName-AbCdEf"
    },
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetRandomPassword"
            ],
            "Resource": "*"
        },
        {
            "Action": [
                "ec2:CreateNetworkInterface",
                "ec2:DeleteNetworkInterface",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DetachNetworkInterface"
            ],
            "Resource": "*",
            "Effect": "Allow"
        }
    ]
}
```

Pernyataan kebijakan untuk kunci yang dikelola pelanggan

Jika rahasia dienkripsi dengan kunci KMS selain Kunci yang dikelola AWS `aws/secretsmanager`, maka Anda perlu memberikan izin peran eksekusi Lambda untuk menggunakan kunci tersebut. Anda dapat menggunakan konteks enkripsi secretArn untuk membatasi penggunaan fungsi dekripsi, sehingga peran fungsi rotasi hanya memiliki akses untuk mendekripsi rahasia yang bertanggung jawab untuk berputar. Contoh berikut menunjukkan pernyataan untuk ditambahkan ke kebijakan peran eksekusi untuk mendekripsi rahasia menggunakan kunci KMS.

```
    {
        "Effect": "Allow",
        "Action": [
            "kms:Decrypt",
            "kms:DescribeKey",
            "kms:GenerateDataKey"
        ],
        "Resource": "KMSKeyARN",
        "Condition": {
```

```
                "StringEquals": {
                    "kms:EncryptionContext:SecretARN": "SecretARN"
                }
            }
        }
```

Untuk menggunakan fungsi rotasi untuk beberapa rahasia yang dienkripsi dengan kunci yang dikelola pelanggan, tambahkan pernyataan seperti contoh berikut untuk memungkinkan peran eksekusi mendekripsi rahasia.

```
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt",
                "kms:DescribeKey",
                "kms:GenerateDataKey"
            ],
            "Resource": "KMSKeyARN",
            "Condition": {
                "StringEquals": {
                    "kms:EncryptionContext:SecretARN": [
                        "arn1",
                        "arn2"
                    ]
                }
            }
        }
```

Pernyataan kebijakan untuk strategi pengguna bergantian

Untuk informasi tentang strategi rotasi pengguna bergantian, lihatthe section called “Strategi rotasi fungsi Lambda”.

Untuk rahasia yang berisi kredensi Amazon RDS, jika Anda menggunakan strategi pengguna bergantian dan rahasia pengguna super dikelola oleh Amazon RDS, maka Anda juga harus mengizinkan fungsi rotasi untuk memanggil API hanya-baca di Amazon RDS sehingga bisa mendapatkan informasi koneksi untuk database. Kami menyarankan Anda melampirkan kebijakan AWS terkelola AmazonRDSReadOnlyAccess.

Contoh kebijakan berikut memungkinkan fungsi untuk:

- Jalankan operasi Secrets Manager untuk*SecretARN*.

- Ambil kredensialnya di rahasia superuser. Secrets Manager menggunakan kredensil dalam rahasia superuser untuk memperbarui kredensil dalam rahasia yang diputar.
- Buat kata sandi baru.
- Siapkan konfigurasi yang diperlukan jika database atau layanan Anda berjalan di VPC. Untuk informasi selengkapnya, lihat Mengonfigurasi fungsi Lambda untuk mengakses sumber daya di VPC.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "secretsmanager:UpdateSecretVersionStage"
            ],
            "Resource": "arn:aws:secretsmanager:us-
east-1:123456789012:secret:secretName-AbCdEf"
    },
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue"
            ],
            "Resource": "arn:aws:secretsmanager:us-
east-1:123456789012:secret:secretName-AbCdEf"
    },
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetRandomPassword"
            ],
            "Resource": "*"
        },
        {
            "Action": [
```

```
                "ec2:CreateNetworkInterface",
                "ec2:DeleteNetworkInterface",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DetachNetworkInterface"
            ],
            "Resource": "*",
            "Effect": "Allow"
        }
    ]
}
```

Akses jaringan untuk fungsi AWS Lambda rotasi

Karenathe section called “Rotasi dengan fungsi Lambda”, ketika Secrets Manager menggunakan fungsi Lambda untuk memutar rahasia, fungsi rotasi Lambda harus dapat mengakses rahasia. Jika rahasia Anda berisi kredensi, maka fungsi Lambda juga harus dapat mengakses sumber kredensional tersebut, seperti database atau layanan.

Untuk mengakses rahasia

- Fungsi rotasi Lambda Anda harus dapat mengakses titik akhir Secrets Manager. Jika fungsi Lambda Anda dapat mengakses internet, maka Anda dapat menggunakan titik akhir publik. Untuk menemukan titik akhir, lihatthe section called “Titik akhir Manajer Rahasia”.

- Jika fungsi Lambda Anda berjalan di VPC yang tidak memiliki akses internet, kami sarankan Anda mengonfigurasi titik akhir pribadi layanan Secrets Manager dalam VPC Anda. VPC Anda kemudian dapat mencegat permintaan yang ditujukan ke titik akhir regional publik dan mengarahkannya ke titik akhir pribadi. Untuk informasi selengkapnya, lihat Titik akhir VPC (AWS PrivateLink).

- Atau, Anda dapat mengaktifkan fungsi Lambda Anda untuk mengakses titik akhir publik Secrets Manager dengan menambahkan gateway NAT atau gateway internet ke VPC Anda, yang memungkinkan lalu lintas dari VPC Anda mencapai titik akhir publik. Ini membuat VPC Anda berisiko lebih besar karena alamat IP untuk gateway dapat diserang dari Internet publik.

(Opsional) Untuk mengakses database atau layanan

- Untuk rahasia seperti kunci API, tidak ada database sumber atau layanan yang perlu Anda perbarui bersama dengan rahasianya.

Jika database atau layanan Anda berjalan pada instans Amazon EC2 di VPC, sebaiknya Anda mengonfigurasi fungsi Lambda agar berjalan di VPC yang sama. Kemudian fungsi rotasi dapat berkomunikasi langsung dengan layanan Anda. Untuk informasi selengkapnya, lihat Mengonfigurasi akses VPC.

Untuk mengizinkan fungsi Lambda mengakses database atau layanan, Anda harus memastikan bahwa grup keamanan yang dilampirkan ke fungsi rotasi Lambda Anda memungkinkan koneksi keluar ke database atau layanan. Anda juga harus memastikan bahwa grup keamanan yang dilampirkan ke database atau layanan Anda mengizinkan koneksi masuk dari fungsi rotasi Lambda.

Pemecahan Masalah AWS Secrets Manager rotasi

Untuk banyak layanan, Secrets Manager menggunakan fungsi Lambda untuk memutar rahasia. Untuk informasi selengkapnya, lihat the section called “Rotasi dengan fungsi Lambda”. Fungsi rotasi Lambda berinteraksi dengan database atau layanan rahasianya serta Secrets Manager. Ketika rotasi tidak bekerja seperti yang Anda harapkan, Anda harus terlebih dahulu memeriksa CloudWatch log.

> **Note**
>
> Beberapa layanan dapat mengelola rahasia untuk Anda, termasuk mengelola rotasi otomatis. Untuk informasi selengkapnya, lihat the section called “Rotasi terkelola”.

Topik

- Cara memecahkan masalah kegagalan rotasi rahasia di AWS Lambda fungsi
- Tidak ada aktivitas setelah “Menemukan kredensyal dalam variabel lingkungan”
- Tidak ada aktivitas setelah “createSecret”
- Kesalahan: “Akses ke KMS tidak diizinkan”
- Kesalahan: “Kunci hilang dari JSON rahasia”
- Kesalahan: “setSecret: Tidak dapat masuk ke database”
- Kesalahan: “Tidak dapat mengimpor modul 'lambda_function'”
- Tingkatkan fungsi rotasi yang ada ke Python 3.12
- AWS Lambda rotasi rahasia dengan PutSecretValuegagal

- Kesalahan: “Kesalahan saat menjalankan lambda <arn > selama <rotasi > langkah”

Cara memecahkan masalah kegagalan rotasi rahasia di AWS Lambda fungsi

Jika Anda mengalami kegagalan rotasi rahasia dengan fungsi Lambda Anda, gunakan langkah-langkah berikut untuk memecahkan masalah dan menyelesaikan masalah.

Kemungkinan penyebab

- Eksekusi bersamaan yang tidak memadai untuk fungsi Lambda
- Kondisi balapan karena beberapa panggilan API selama rotasi
- Logika fungsi Lambda salah
- Masalah jaringan antara fungsi Lambda dan database

Langkah-langkah pemecahan masalah umum

1. Menganalisis CloudWatch log:
   - Cari pesan kesalahan tertentu atau perilaku tak terduga di log fungsi Lambda
   - Verifikasi bahwa semua langkah rotasi (CreateSecretSetSecret,TestSecret,,FinishSecret) sedang dicoba
2. Tinjau panggilan API selama rotasi:
   - Hindari membuat panggilan API yang bermutasi pada rahasia selama rotasi Lambda
   - Pastikan tidak ada kondisi balapan antara RotateSecret dan PutSecretValue panggilan
3. Verifikasi logika fungsi Lambda:
   - Konfirmasikan bahwa Anda menggunakan kode AWS sampel terbaru untuk rotasi rahasia
   - Jika menggunakan kode khusus, tinjau untuk penanganan yang tepat dari semua langkah rotasi
4. Periksa konfigurasi jaringan:
   - Verifikasi aturan grup keamanan memungkinkan fungsi Lambda mengakses database
   - Pastikan titik akhir VPC atau akses titik akhir publik yang tepat untuk Secrets Manager
5. Uji versi rahasia:
   - Verifikasi bahwa versi rahasia AWSCURRENT memungkinkan akses database

- Periksa apakah versi AWSPREVIOUS atau AWSPENDING valid

6. Hapus rotasi yang tertunda:

   - Jika rotasi gagal secara konsisten, kosongkan label pementasan AWSPENDING dan coba lagi rotasi

7. Periksa pengaturan konkurensi Lambda:

   - Verifikasi bahwa pengaturan konkurensi sesuai untuk beban kerja Anda
   - Jika Anda mencurigai adanya masalah konkurensi, lihat bagian “Mengatasi masalah kegagalan rotasi terkait kesepakatan”

Tidak ada aktivitas setelah “Menemukan kredensyal dalam variabel lingkungan”

Jika tidak ada aktivitas setelah “Menemukan kredensyal dalam variabel lingkungan”, dan durasi tugas panjang, misalnya batas waktu Lambda default 30000ms, maka fungsi Lambda mungkin habis waktu saat mencoba mencapai titik akhir Secrets Manager.

Fungsi rotasi Lambda Anda harus dapat mengakses titik akhir Secrets Manager. Jika fungsi Lambda Anda dapat mengakses internet, maka Anda dapat menggunakan titik akhir publik. Untuk menemukan titik akhir, lihatthe section called “Titik akhir Manajer Rahasia”.

Jika fungsi Lambda Anda berjalan di VPC yang tidak memiliki akses internet, kami sarankan Anda mengonfigurasi titik akhir pribadi layanan Secrets Manager dalam VPC Anda. VPC Anda kemudian dapat mencegat permintaan yang ditujukan ke titik akhir regional publik dan mengarahkannya ke titik akhir pribadi. Untuk informasi selengkapnya, lihat Titik akhir VPC (AWS PrivateLink).

Atau, Anda dapat mengaktifkan fungsi Lambda Anda untuk mengakses titik akhir publik Secrets Manager dengan menambahkan gateway NAT atau gateway internet ke VPC Anda, yang memungkinkan lalu lintas dari VPC Anda mencapai titik akhir publik. Ini membuat VPC Anda berisiko lebih besar karena alamat IP untuk gateway dapat diserang dari Internet publik.

Tidak ada aktivitas setelah “createSecret”

Berikut ini adalah masalah yang dapat menyebabkan rotasi berhenti setelah createSecret:

ACL Jaringan VPC tidak mengizinkan lalu lintas HTTPS masuk dan keluar.

Untuk informasi selengkapnya, lihat Mengontrol lalu lintas ke subnet menggunakan ACL Jaringan di Panduan Pengguna Amazon VPC.

Konfigurasi batas waktu fungsi Lambda terlalu pendek untuk melakukan tugas.

Untuk informasi selengkapnya, lihat Mengonfigurasi opsi fungsi Lambda di Panduan AWS Lambda Pengembang.

Titik akhir VPC Secrets Manager tidak mengizinkan CIDR VPC saat masuk ke dalam grup keamanan yang ditetapkan.

Untuk informasi selengkapnya, lihat Mengontrol lalu lintas ke sumber daya menggunakan grup keamanan di Panduan Pengguna Amazon VPC.

Kebijakan titik akhir VPC Secrets Manager tidak mengizinkan Lambda menggunakan titik akhir VPC.

Untuk informasi selengkapnya, lihat the section called “Titik akhir VPC (AWS PrivateLink)”.

Rahasianya menggunakan rotasi pengguna bergantian, rahasia superuser dikelola oleh Amazon RDS, dan fungsi Lambda tidak dapat mengakses RDS API.

Untuk rotasi pengguna bergantian di mana rahasia superuser dikelola oleh AWS layanan lain, fungsi rotasi Lambda harus dapat memanggil titik akhir layanan untuk mendapatkan informasi koneksi database. Kami menyarankan Anda mengonfigurasi titik akhir VPC untuk layanan database. Untuk informasi lebih lanjut, lihat:

- Amazon RDS API dan titik akhir VPC antarmuka di Panduan Pengguna Amazon RDS.
- Bekerja dengan titik akhir VPC di Panduan Manajemen Pergeseran Merah Amazon.

Kesalahan: “Akses ke KMS tidak diizinkan”

Jika Anda lihat`ClientError: An error occurred (AccessDeniedException) when calling the GetSecretValue operation: Access to KMS is not allowed`, fungsi rotasi tidak memiliki izin untuk mendekripsi rahasia menggunakan kunci KMS yang digunakan untuk mengenkripsi rahasia. Mungkin ada kondisi dalam kebijakan izin yang membatasi konteks enkripsi ke rahasia tertentu. Untuk informasi tentang izin yang diperlukan, lihatthe section called “Pernyataan kebijakan untuk kunci yang dikelola pelanggan”.

Kesalahan: “Kunci hilang dari JSON rahasia”

Fungsi rotasi Lambda membutuhkan nilai rahasia berada dalam struktur JSON tertentu. Jika Anda melihat kesalahan ini, maka JSON mungkin kehilangan kunci yang coba diakses oleh fungsi rotasi. Untuk informasi tentang struktur JSON untuk setiap jenis rahasia, lihatthe section called “Struktur JSON dari sebuah rahasia”.

Kesalahan: “setSecret: Tidak dapat masuk ke database”

Berikut ini adalah masalah yang dapat menyebabkan kesalahan ini:

Fungsi rotasi tidak dapat mengakses database.

Jika durasi tugas panjang, misalnya lebih dari 5000 ms, maka fungsi rotasi Lambda mungkin tidak dapat mengakses database melalui jaringan.

Jika database atau layanan Anda berjalan pada instans Amazon EC2 di VPC, sebaiknya Anda mengonfigurasi fungsi Lambda agar berjalan di VPC yang sama. Kemudian fungsi rotasi dapat berkomunikasi langsung dengan layanan Anda. Untuk informasi selengkapnya, lihat Mengonfigurasi akses VPC.

Untuk mengizinkan fungsi Lambda mengakses database atau layanan, Anda harus memastikan bahwa grup keamanan yang dilampirkan ke fungsi rotasi Lambda Anda memungkinkan koneksi keluar ke database atau layanan. Anda juga harus memastikan bahwa grup keamanan yang dilampirkan ke database atau layanan Anda mengizinkan koneksi masuk dari fungsi rotasi Lambda.

Kredensi dalam rahasia tidak benar.

Jika durasi tugas pendek, maka fungsi rotasi Lambda mungkin tidak dapat mengautentikasi dengan kredensyal dalam rahasia. Periksa kredensyal dengan masuk secara manual dengan informasi dalam `AWSCURRENT` dan `AWSPREVIOUS` versi rahasia menggunakan perintah.AWS CLI`get-secret-value`

Database digunakan **`scram-sha-256`** untuk mengenkripsi kata sandi.

Jika database Anda adalah Aurora PostgreSQL versi 13 atau yang lebih baru dan digunakan `scram-sha-256` untuk mengenkripsi kata sandi, tetapi fungsi rotasi menggunakan `libpq` versi 9 atau lebih lama yang tidak mendukung`scram-sha-256`, maka fungsi rotasi tidak dapat terhubung ke database.

Untuk menentukan pengguna database mana yang menggunakan enkripsi **`scram-sha-256`**

- Lihat Memeriksa pengguna dengan kata sandi non-Scram di blog Otentikasi SCRAM di RDS untuk PostgreSQL 13.

Untuk menentukan versi **libpq** yang digunakan fungsi rotasi Anda

1. Di Linux-based komputer, di konsol Lambda, navigasikan ke fungsi rotasi Anda dan unduh bundel penerapan. Buka kompres file zip ke direktori kerja.
2. Pada baris perintah, di direktori kerja, jalankan:

   ```
   readelf -a libpq.so.5 | grep RUNPATH
   ```
3. Jika Anda melihat string*PostgreSQL-9.4.x*, atau versi utama kurang dari 10, maka fungsi rotasi tidak mendukung`scram-sha-256`.
   - Output untuk fungsi rotasi yang tidak mendukung`scram-sha-256`:

     ```
     0x000000000000001d (RUNPATH) Library runpath: [/
     local/p4clients/pkgbuild-a1b2c/workspace/build/
     PostgreSQL/PostgreSQL-9.4.x_client_only.123456.0/AL2_x86_64/
     DEV.STD.PTHREAD/build/private/tmp/brazil-path/build.libfarm/lib:/
     local/p4clients/pkgbuild-a1b2c/workspace/src/PostgreSQL/build/
     private/install/lib]
     ```
   - Output untuk fungsi rotasi yang mendukung`scram-sha-256`:

     ```
     0x000000000000001d (RUNPATH) Library runpath: [/
     local/p4clients/pkgbuild-a1b2c/workspace/build/
     PostgreSQL/PostgreSQL-10.x_client_only.123456.0/AL2_x86_64/
     DEV.STD.PTHREAD/build/private/tmp/brazil-path/build.libfarm/lib:/
     local/p4clients/pkgbuild-a1b2c/workspace/src/PostgreSQL/build/
     private/install/lib]
     ```
   - Output untuk fungsi rotasi yang mendukung`scram-sha-256`:

     ```
     0x000000000000001d (RUNPATH) Library runpath: [/local/
     p4clients/pkgbuild- a1b2c /workspace/build/PostgreSQL/
     PostgreSQL-14.x_client_only. 123456 .0/AL2_x86_64/
     DEV.STD.PTHREAD/build/private/tmp/brazil-path/build.libfarm/lib:/
     local/p4clients/pkgbuild- a1b2c /workspace/src/PostgreSQL/build/
     private/install/lib]
     ```
   - Output untuk fungsi rotasi yang mendukung`scram-sha-256`:

     ```
     0x000000000000001d (RUNPATH) Library runpath: [/local/p4clients/
     pkgbuild- a1b2c/workspace/build/PostgreSQL/PostgreSQL-
     ```

```
14.x_client_only.123456.0/AL2_x86_64/DEV.STD.PTHREAD/build/
private/tmp/brazil- path/build.libfarm/lib:/local/p4clients/
pkgbuild- a1b2c/workspace/src/PostgreSQL/build/private/install/
lib]
```

> **Note**
>
> Jika Anda mengatur rotasi rahasia otomatis sebelum 30 Desember 2021, fungsi rotasi Anda menggabungkan versi sebelumnya `libpq` yang tidak mendukung`scram-sha-256`. Untuk mendukung`scram-sha-256`, Anda perlu membuat ulang fungsi rotasi Anda.

Database membutuhkan SSL/TLS akses.

Jika database Anda memerlukan SSL/TLS koneksi, tetapi fungsi rotasi menggunakan koneksi yang tidak terenkripsi, maka fungsi rotasi tidak dapat terhubung ke database. Fungsi rotasi untuk Amazon RDS (kecuali Oracle dan Db2) dan Amazon DocumentDB secara otomatis menggunakan Secure Socket Layer (SSL) atau Transport Layer Security (TLS) untuk terhubung ke database Anda, jika tersedia. Jika tidak, mereka menggunakan koneksi yang tidak terenkripsi.

> **Note**
>
> Jika Anda mengatur rotasi rahasia otomatis sebelum 20 Desember 2021, fungsi rotasi Anda mungkin didasarkan pada templat sebelumnya yang tidak mendukung SSL/TLS. Untuk mendukung koneksi yang digunakan SSL/TLS, Anda perlu membuat ulang fungsi rotasi Anda.

Untuk menentukan kapan fungsi rotasi Anda dibuat

1. Di konsol Secrets Manager https://console.aws.amazon.com/secretsmanager/, buka rahasia Anda. Di bagian konfigurasi Rotasi, di bawah fungsi rotasi Lambda, Anda melihat fungsi Lambda ARN, misalnya,. `arn:aws:lambda:`*`aws-region`*`:`*`123456789012`*`:function:`*`SecretsManagerMyRotationFunction`* Salin nama fungsi dari akhir ARN, dalam contoh ini. *`SecretsManagerMyRotationFunction`*
2. Di AWS Lambda konsol https://console.aws.amazon.com/lambda/, di bawah Fungsi, tempel nama fungsi Lambda Anda di kotak pencarian, pilih Enter, lalu pilih fungsi Lambda.

3. Di halaman detail fungsi, pada tab Konfigurasi, di bawah Tag, salin nilai di sebelah kunci aws:cloudformation:stack-name.
4. Di AWS CloudFormation konsol https://console.aws.amazon.com/cloudformation, di bawah Tumpukan, tempel nilai kunci di kotak pencarian, lalu pilih Enter.
5. Daftar tumpukan menyaring sehingga hanya tumpukan yang membuat fungsi rotasi Lambda yang muncul. Di kolom Tanggal dibuat, lihat tanggal tumpukan dibuat. Ini adalah tanggal fungsi rotasi Lambda dibuat.

Kesalahan: “Tidak dapat mengimpor modul 'lambda_function'”

Anda mungkin menerima kesalahan ini jika Anda menjalankan fungsi Lambda sebelumnya yang secara otomatis ditingkatkan dari Python 3.7 ke versi Python yang lebih baru. Untuk mengatasi kesalahan, Anda dapat mengubah versi fungsi Lambda kembali ke Python 3.7, dan kemudian. the section called “Tingkatkan fungsi rotasi yang ada dari Python 3.7 ke Python 3.12” Untuk informasi selengkapnya, lihat Mengapa rotasi fungsi Secrets Manager Lambda saya gagal dengan kesalahan “modul pg tidak ditemukan”? di AWS re:post.

Tingkatkan fungsi rotasi yang ada ke Python 3.12

Secrets Manager sedang beralih ke Python 3.12 untuk fungsi rotasi Lambda. Untuk beralih ke fungsi rotasi baru yang menggunakan Python 3.12, Anda harus mengikuti jalur pemutakhiran berdasarkan metode penerapan dan rotasi lambda versi Python saat ini. Gunakan prosedur berikut untuk memutakhirkan versi Python dan dependensi yang mendasarinya.

Tingkatkan fungsi rotasi yang ada dari Python 3.7 ke Python 3.12

Beberapa fungsi rotasi yang dibuat sebelum November 2022 menggunakan Python 3.7.AWS SDK untuk Python berhenti mendukung Python 3.7 pada Desember 2023. Untuk informasi selengkapnya, lihat Pembaruan kebijakan dukungan Python untuk AWS SDK dan Alat. Untuk beralih ke fungsi rotasi baru yang menggunakan Python 3.12, Anda dapat menambahkan properti runtime ke fungsi rotasi yang ada atau membuat ulang fungsi rotasi.

Untuk menemukan fungsi rotasi Lambda mana yang menggunakan Python 3.7

1. Masuk ke Konsol Manajemen AWS dan buka AWS Lambda konsol di https://console.aws.amazon.com/lambda/.
2. Dalam daftar Fungsi, filter untuk**`SecretsManager`**.
3. Dalam daftar fungsi yang difilter, di bawah Runtime, cari Python 3.7.

Untuk meningkatkan ke Python 3.12:

- Opsi 1: Buat ulang fungsi rotasi menggunakan CloudFormation
- Opsi 2: Perbarui runtime untuk fungsi rotasi yang ada menggunakan CloudFormation
- Opsi 3: Untuk AWS CDK pengguna, tingkatkan perpustakaan CDK

Opsi 1: Buat ulang fungsi rotasi menggunakan CloudFormation

Saat Anda menggunakan konsol Secrets Manager untuk mengaktifkan rotasi, Secrets Manager menggunakan CloudFormation untuk membuat sumber daya yang diperlukan, termasuk fungsi rotasi Lambda. Jika Anda menggunakan konsol untuk mengaktifkan rotasi, atau Anda membuat fungsi rotasi menggunakan CloudFormation tumpukan, Anda dapat menggunakan CloudFormation tumpukan yang sama untuk membuat ulang fungsi rotasi dengan nama baru. Fungsi baru menggunakan versi Python yang lebih baru.

Untuk menemukan CloudFormation tumpukan yang menciptakan fungsi rotasi

- Pada halaman detail fungsi Lambda, pada tab Konfigurasi, pilih Tag. Lihat ARN di sebelah aws:cloudformation:stack-id.

  Nama tumpukan disematkan di ARN, seperti yang ditunjukkan pada contoh berikut.

  - ARN: `arn:aws:cloudformation:`*`aws-region`*`:123456789012:stack/`**`SecretsManagerRDSMySQLRotationSingleUser5c2-SecretRotationScheduleHostedRotationLambda`**`-3CUDHZMDMBO8/79fc9050-2eef-11ed-`
  - Nama tumpukan: **`SecretsManagerRDSMySQLRotationSingleUser5c2-SecretRotationScheduleHostedRotationLambda`**

Untuk membuat ulang fungsi rotasi (CloudFormation)

1. Di CloudFormation, cari tumpukan berdasarkan nama, lalu pilih Perbarui.

   Jika kotak dialog muncul merekomendasikan Anda memperbarui tumpukan root, pilih Buka tumpukan root, lalu pilih Perbarui.
2. Pada halaman Update stack, di bawah Siapkan template, pilih Edit di Application Composer, dan kemudian di bawah Edit template di Application Composer, pilih tombol Edit di Application Composer.
3. Di Application Composer, lakukan hal berikut:

a. Dalam kode template, di`SecretRotationScheduleHostedRotationLambda`, ganti nilai untuk `"functionName": "SecretsManagerTestRotationRDS"` dengan nama fungsi baru, misalnya di JSON, **`"functionName": "SecretsManagerTestRotationRDSupdated"`**

b. Pilih Perbarui template.

c. Di kotak CloudFormation dialog Lanjutkan ke, pilih Konfirmasi dan lanjutkan ke CloudFormation.

4. Lanjutkan melalui alur kerja CloudFormation tumpukan dan kemudian pilih Kirim.

Opsi 2: Perbarui runtime untuk fungsi rotasi yang ada menggunakan CloudFormation

Saat Anda menggunakan konsol Secrets Manager untuk mengaktifkan rotasi, Secrets Manager menggunakan CloudFormation untuk membuat sumber daya yang diperlukan, termasuk fungsi rotasi Lambda. Jika Anda menggunakan konsol untuk mengaktifkan rotasi, atau Anda membuat fungsi rotasi menggunakan CloudFormation tumpukan, Anda dapat menggunakan CloudFormation tumpukan yang sama untuk memperbarui runtime untuk fungsi rotasi.

Untuk menemukan CloudFormation tumpukan yang menciptakan fungsi rotasi

- Pada halaman detail fungsi Lambda, pada tab Konfigurasi, pilih Tag. Lihat ARN di sebelah aws:cloudformation:stack-id.

  Nama tumpukan disematkan di ARN, seperti yang ditunjukkan pada contoh berikut.

  - ARN: `arn:aws:cloudformation:`*`aws-region`*`:123456789012:stack/`**`SecretsManagerRDSMySQLRotationSingleUser5c2-SecretRotationScheduleHostedRotationLambda`**`-3CUDHZMDMBO8/79fc9050-2eef-11ed-`
  - Nama tumpukan: **`SecretsManagerRDSMySQLRotationSingleUser5c2-SecretRotationScheduleHostedRotationLambda`**

Untuk memperbarui runtime untuk fungsi rotasi (CloudFormation)

1. Di CloudFormation, cari tumpukan berdasarkan nama, lalu pilih Perbarui.

   Jika kotak dialog muncul merekomendasikan Anda memperbarui tumpukan root, pilih Buka tumpukan root, lalu pilih Perbarui.

2. Pada halaman Update stack, di bawah Siapkan template, pilih Edit di Application Composer, dan kemudian di bawah Edit template di Application Composer, pilih tombol Edit di Application Composer.
3. Di Application Composer, lakukan hal berikut:
   a. Dalam template JSON, untuk, di bawah`SecretRotationScheduleHostedRotationLambda`, di bawah `PropertiesParameters`, tambahkan**`"runtime": "python3.12"`**.
   b. Pilih Perbarui template.
   c. Di kotak CloudFormation dialog Lanjutkan ke, pilih Konfirmasi dan lanjutkan ke CloudFormation.
4. Lanjutkan melalui alur kerja CloudFormation tumpukan dan kemudian pilih Kirim.

Opsi 3: Untuk AWS CDK pengguna, tingkatkan perpustakaan CDK

Jika Anda menggunakan versi AWS CDK sebelumnya v2.258.0 untuk mengatur rotasi rahasia Anda, Anda dapat memperbarui fungsi Lambda ke Python 3.12 dengan memutakhirkan ke v2.258.0 atau yang lebih baru.AWS CDK Untuk informasi selengkapnya, lihat Panduan Pengembang AWS Cloud Development Kit (AWS CDK) v2.

Tingkatkan fungsi rotasi yang ada dari Python 3.9 atau yang lebih baru ke Python 3.12

Untuk menemukan fungsi rotasi Lambda mana yang menggunakan versi Python yang lebih lama

1. Masuk ke Konsol Manajemen AWS dan buka AWS Lambda konsol di https://console.aws.amazon.com/lambda/.
2. Dalam daftar Fungsi, filter untuk**`SecretsManager`**.
3. Dalam daftar fungsi yang difilter, di bawah Runtime, cari versi Python sebelumnya. **`Python 3.12`**

Perbarui jalur dengan metode penerapan

Fungsi rotasi Lambda yang diidentifikasi dalam daftar ini dapat digunakan melalui konsol Secrets Manager,AWS Serverless Application Repository aplikasi, atau transformasi.CloudFormation Masing-masing strategi penyebaran ini memiliki jalur pembaruan yang berbeda.

Gunakan salah satu prosedur berikut untuk memperbarui fungsi rotasi Lambda Anda, tergantung pada bagaimana fungsi Anda diterapkan.

AWS Secrets Manager console-deployed functions

Fungsi Lambda baru harus diterapkan melalui AWS Secrets Manager konsol karena Anda tidak dapat memperbarui dependensi secara manual untuk fungsi Lambda yang ada.

Gunakan prosedur berikut untuk memutakhirkan fungsi yang digunakan AWS Secrets Manager konsol.

1. Buka konsol Secrets Manager di https://console.aws.amazon.com/secretsmanager/.
2. Di bawah AWS Secrets Manager, pilih Rahasia. Pilih rahasia yang menggunakan fungsi Lambda yang ingin Anda perbarui.
3. Arahkan ke tab Rotasi dan pilih opsi Perbarui konfigurasi rotasi.
4. Di bawah Fungsi rotasi, pilih Buat fungsi baru, dan masukkan nama baru untuk fungsi rotasi Lambda.

    a. (Opsional) Setelah pembaruan selesai, Anda dapat menguji fungsi Lambda yang diperbarui untuk mengonfirmasi bahwa pembaruan berfungsi seperti yang diharapkan. Di bawah tab Rotasi, pilih Putar Rahasia Segera untuk memulai rotasi langsung.

    b. (Opsional) Anda dapat melihat log fungsi dan versi Python yang digunakan saat runtime di Amazon. CloudWatch Untuk informasi selengkapnya, lihat Melihat CloudWatch Log untuk fungsi Lambda di Panduan AWS Lambda Pengembang.
5. Setelah fungsi rotasi baru diatur, Anda dapat menghapus fungsi rotasi lama.

AWS Serverless Application Repository deployments

Prosedur berikut menunjukkan cara memutakhirkan AWS Serverless Application Repository penerapan. Fungsi Lambda yang digunakan melalui AWS Serverless Application Repository memiliki spanduk yang menyatakan `This function belongs to an application. Click here to manage it.` yang mencakup tautan ke aplikasi Lambda tempat fungsi tersebut berada.

> **Important**
>
> AWS Serverless Application Repository ketersediaan Wilayah AWS tergantung.

Gunakan prosedur berikut untuk memperbarui fungsi yang AWS Serverless Application Repository diterapkan.

1. Buka AWS Lambda konsol di https://console.aws.amazon.com/lambda/.
2. Arahkan ke tab Konfigurasi fungsi Lambda yang perlu diperbarui.
    - Anda akan memerlukan informasi berikut tentang fungsi Anda saat memperbarui AWS Serverless Application Repository aplikasi yang diterapkan. Anda dapat menemukan informasi ini di konsol Lambda.
        - Nama aplikasi Lambda
            - Nama aplikasi Lambda dapat ditemukan dengan menggunakan tautan di spanduk. Misalnya, spanduk menyatakan hal berikut`serverlessrepo-`*`SecretsManagerRedshiftRotationSingleUser`*. Nama dalam contoh ini adalah`SecretsManagerRedshiftRotationSingleUser`.
        - Nama fungsi rotasi Lambda
        - Titik akhir Secrets Manager
            - Titik akhir dapat ditemukan di bawah tab Konfigurasi dan variabel Lingkungan yang ditetapkan ke variabel SECRETS_MANAGER_ENDPOINT.
3. Untuk meng-upgrade Python, Anda harus memperbarui versi semantik dari aplikasi tanpa server. Lihat Memperbarui Aplikasi di Panduan AWS Serverless Application Repository Pengembang.

Custom Lambda rotation functions

Jika Anda membuat fungsi rotasi Lambda kustom, Anda perlu memutakhirkan setiap dependensi paket dan runtime untuk fungsi-fungsi ini. Untuk informasi selengkapnya, lihat Upgrade runtime fungsi Lambda ke versi terbaru.

AWS::SecretsManager-2024-09-16 transform macro

Jika fungsi Lambda diterapkan melalui transformasi ini, memperbarui tumpukan menggunakan template yang ada akan memungkinkan Anda menggunakan runtime Lambda yang diperbarui.

Gunakan prosedur berikut untuk memperbarui CloudFormation tumpukan menggunakan template yang ada.

1. Buka CloudFormation konsol di https://console.aws.amazon.com/cloudformation.
2. Pada halaman Stacks, pilih tumpukan yang ingin Anda perbarui.
3. Pilih Perbarui pada panel detail tumpukan.
4. Untuk Pilih metode pembaruan templat, pilih Pembaruan langsung.
5. Pada halaman Tentukan template, pilih Gunakan template yang ada.
6. Simpan semua opsi lain pada nilai defaultnya, lalu pilih Perbarui tumpukan.

Jika Anda mengalami masalah saat memperbarui tumpukan, lihat Menentukan penyebab kegagalan tumpukan di Panduan CloudFormation Pengguna.

AWS::SecretsManager-2020-07-23 transform macro

Kami menyarankan Anda bermigrasi ke versi transformasi yang lebih baru jika Anda menggunakan. `AWS::SecretsManager-2020-07-23` Lihat Memperkenalkan versi AWS Secrets Manager transformasi yang disempurnakan:AWS:: SecretsManager-2024-09-16 di Blog AWS Keamanan untuk informasi selengkapnya. Jika Anda terus menggunakan`AWS::SecretsManager-2020-07-23`, Anda dapat mengalami kesalahan ketidakcocokan antara versi runtime Anda dan artefak kode fungsi Lambda. Untuk informasi selengkapnya, lihat AWS::SecretsManager:: RotationSchedule HostedRotationLambda di Referensi CloudFormation Template.

Jika Anda mengalami masalah saat memperbarui tumpukan, Tentukan penyebab kegagalan tumpukan di Panduan CloudFormation Pengguna.

Verifikasi peningkatan Python

Untuk memverifikasi peningkatan Python, buka konsol Lambda (https://console.aws.amazon.com/lambda/) dan akses halaman Fungsi. Pilih fungsi yang Anda perbarui. Di bawah bagian Sumber kode, tinjau file yang disertakan dalam direktori dan pastikan file Python .so adalah versi. `3.12`

AWS Lambda rotasi rahasia dengan **`PutSecretValue`**gagal

Jika Anda menggunakan peran yang diasumsikan atau rotasi lintas akun dengan Secrets Manager dan Anda menemukan RotationFailed acara CloudTrail dengan pesan: Versi rahasia tertunda *`VERSION_ID`* untuk Rahasia tidak *`SECRET_ARN`* dibuat oleh *`LAMBDA_ARN`*. Lambda Hapus `AWSPENDING` label pementasan dan mulai ulang rotasi, maka Anda perlu memperbarui fungsi Lambda Anda untuk menggunakan parameter. `RotationToken`

Perbarui fungsi rotasi Lambda untuk disertakan **`RotationToken`**

1. Unduh kode fungsi Lambda

   - Buka konsol Lambda
   - Di panel navigasi, pilih Fungsi
   - Pilih fungsi rotasi rahasia Lambda Anda untuk nama Fungsi
   - Untuk Download, pilih salah satu Kode fungsi.zip, AWS SAM file, Keduanya
   - Pilih OK untuk menyimpan fungsi pada mesin lokal Anda.

2. Sunting `Lambda_handler`

   Sertakan parameter rotation_token dalam langkah create_secret untuk rotasi lintas akun:

```
def lambda_handler(event, context):
    """Secrets Manager Rotation Template

    This is a template for creating an AWS Secrets Manager rotation lambda

    Args:
        event (dict): Lambda dictionary of event parameters. These keys must
 include the following:
            - SecretId: The secret ARN or identifier
            - ClientRequestToken: The ClientRequestToken of the secret version
            - Step: The rotation step (one of createSecret, setSecret, testSecret,
 or finishSecret)
            - RotationToken: the rotation token to put as parameter for
 PutSecretValue call

        context (LambdaContext): The Lambda runtime information

    Raises:
        ResourceNotFoundException: If the secret with the specified arn and stage
 does not exist

        ValueError: If the secret is not properly configured for rotation

        KeyError: If the event parameters do not contain the expected keys

    """
    arn = event['SecretId']
```

```
    token = event['ClientRequestToken']
    step = event['Step']
    # Add the rotation token
    rotation_token = event['RotationToken']

    # Setup the client
    service_client = boto3.client('secretsmanager',
endpoint_url=os.environ['SECRETS_MANAGER_ENDPOINT'])

    # Make sure the version is staged correctly
    metadata = service_client.describe_secret(SecretId=arn)
    if not metadata['RotationEnabled']:
        logger.error("Secret %s is not enabled for rotation" % arn)
        raise ValueError("Secret %s is not enabled for rotation" % arn)
    versions = metadata['VersionIdsToStages']
    if token not in versions:
        logger.error("Secret version %s has no stage for rotation of secret %s." %
(token, arn))
        raise ValueError("Secret version %s has no stage for rotation of secret
%s." % (token, arn))
    if "AWSCURRENT" in versions[token]:
        logger.info("Secret version %s already set as AWSCURRENT for secret %s." %
(token, arn))
        return
    elif "AWSPENDING" not in versions[token]:
        logger.error("Secret version %s not set as AWSPENDING for rotation of
secret %s." % (token, arn))
        raise ValueError("Secret version %s not set as AWSPENDING for rotation of
secret %s." % (token, arn))
    # Use rotation_token
    if step == "createSecret":
        create_secret(service_client, arn, token, rotation_token)

    elif step == "setSecret":
        set_secret(service_client, arn, token)

    elif step == "testSecret":
        test_secret(service_client, arn, token)

    elif step == "finishSecret":
        finish_secret(service_client, arn, token)

    else:
        raise ValueError("Invalid step parameter")
```

3. Edit `create_secret` kode

   Merevisi `create_secret` fungsi untuk menerima dan menggunakan `rotation_token` parameter:

```
# Add rotation_token to the function
def create_secret(service_client, arn, token, rotation_token):
"""Create the secret

This method first checks for the existence of a secret for the passed in token. If
 one does not exist, it will generate a
new secret and put it with the passed in token.

Args:
service_client (client): The secrets manager service client

arn (string): The secret ARN or other identifier

token (string): The ClientRequestToken associated with the secret version

rotation_token (string): the rotation token to put as parameter for PutSecretValue
 call

Raises:
ResourceNotFoundException: If the secret with the specified arn and stage does not
 exist

"""
# Make sure the current secret exists
service_client.get_secret_value(SecretId=arn, VersionStage="AWSCURRENT")

# Now try to get the secret version, if that fails, put a new secret
try:
service_client.get_secret_value(SecretId=arn, VersionId=token,
 VersionStage="AWSPENDING")
logger.info("createSecret: Successfully retrieved secret for %s." % arn)
except service_client.exceptions.ResourceNotFoundException:
# Get exclude characters from environment variable
exclude_characters = os.environ['EXCLUDE_CHARACTERS'] if 'EXCLUDE_CHARACTERS' in
 os.environ else '/@"\'\\'
```

```
# Generate a random password
passwd = service_client.get_random_password(ExcludeCharacters=exclude_characters)

# Put the secret, using rotation_token
service_client.put_secret_value(SecretId=arn, ClientRequestToken=token,
 SecretString=passwd['RandomPassword'], VersionStages=['AWSPENDING'],
 RotationToken=rotation_token)
logger.info("createSecret: Successfully put secret for ARN %s and version %s." %
 (arn, token))
```

4. Unggah kode fungsi Lambda yang diperbarui

   Setelah memperbarui kode fungsi Lambda Anda, unggah untuk memutar rahasia Anda.

Kesalahan: “Kesalahan saat menjalankan lambda *<arn >* selama *<rotasi >* langkah”

Jika Anda mengalami kegagalan rotasi rahasia intermiten dengan fungsi Lambda Anda macet dalam satu lingkaran set, misalnya antara CreateSecret danSetSecret, masalahnya mungkin terkait dengan pengaturan konkurensi.

Langkah pemecahan masalah konkurensi

> **Warning**
>
> Menyetel parameter konkurensi yang disediakan ke nilai yang lebih rendah dari 10 dapat menyebabkan pelambatan karena utas eksekusi yang tidak mencukupi untuk fungsi Lambda. Untuk informasi selengkapnya, lihat Memahami konkurensi cadangan dan konkurensi yang disediakan di Panduan Pengembang.AWS LambdaAWS Lambda

1. Periksa dan sesuaikan pengaturan konkurensi Lambda:
   - Verifikasi `reserved_concurrent_executions` bahwa tidak disetel terlalu rendah (misalnya, 1)
   - Jika menggunakan konkurensi cadangan, setel ke setidaknya 10
   - Pertimbangkan untuk menggunakan konkurensi tanpa syarat untuk lebih banyak fleksibilitas
2. Untuk konkurensi yang disediakan:

- Jangan setel parameter konkurensi yang disediakan secara eksplisit (misalnya, di Terraform).
- Jika Anda harus mengaturnya, gunakan nilai minimal 10.
- Uji secara menyeluruh untuk memastikan nilai yang dipilih berfungsi untuk kasus penggunaan Anda.

3. Pantau dan sesuaikan konkurensi:

- Hitung konkurensi menggunakan rumus ini: Concurrency = (permintaan rata-rata per detik) * (durasi permintaan rata-rata dalam detik). Untuk informasi selengkapnya, lihat Memperkirakan konkurensi cadangan.
- Amati dan catat nilai selama rotasi untuk menentukan pengaturan konkurensi yang sesuai.
- Hati-hati saat menetapkan nilai konkurensi rendah. Mereka dapat menyebabkan pelambatan jika tidak ada cukup utas eksekusi yang tersedia.

Untuk informasi selengkapnya tentang mengonfigurasi konkurensi Lambda, lihat Mengonfigurasi konkurensi cadangan dan Mengonfigurasi konkurensi yang disediakan di Panduan Pengembang.AWS Lambda

Jadwal rotasi

Secrets Manager memutar rahasia Anda pada jadwal selama jendela rotasi yang Anda tetapkan. Untuk mengatur jadwal dan jendela, Anda menggunakan ekspresi cron () atau rate () bersama dengan durasi jendela. Secrets Manager memutar rahasia Anda kapan saja selama jendela rotasi. Anda dapat memutar rahasia sesering setiap empat jam dalam jendela rotasi sekecil satu jam.

Untuk mengaktifkan rotasi, lihat:

- the section called “Rotasi terkelola”
- the section called “Rotasi otomatis untuk rahasia database (konsol)”
- the section called “Rotasi otomatis untuk rahasia non-database (konsol)”

Jadwal rotasi Secrets Manager menggunakan zona waktu UTC.

Jendela rotasi

Jendela rotasi Secrets Manager mirip dengan jendela pemeliharaan. Anda mengatur jendela rotasi ketika Anda ingin rahasia Anda diputar, dan Secrets Manager memutar rahasia Anda pada suatu waktu selama jendela rotasi.

Jendela rotasi Secrets Manager selalu dimulai pada jam. Untuk jadwal rotasi yang menggunakan `rate()` ekspresi dalam hari, jendela rotasi dimulai pada tengah malam. Anda dapat mengatur waktu mulai untuk jendela rotasi dengan menggunakan `cron()` ekspresi. Sebagai contoh, lihat the section called “Ekspresi Cron”.

Secara default, jendela rotasi ditutup setelah satu jam untuk jadwal rotasi dalam jam, dan pada akhir hari untuk jadwal rotasi dalam beberapa hari.

Untuk mengubah panjang jendela rotasi, atur Durasi jendela. Anda dapat mengatur jendela rotasi sekecil satu jam. Jendela rotasi tidak boleh meluas ke jendela rotasi berikutnya. Dengan kata lain, untuk jadwal rotasi dalam jam, konfirmasikan bahwa jendela rotasi kurang dari atau sama dengan jumlah jam antar rotasi. Untuk jadwal rotasi dalam beberapa hari, konfirmasikan bahwa jam mulai ditambah durasi jendela kurang dari atau sama dengan 24 jam.

Ekspresi rate

Ekspresi tingkat Secrets Manager memiliki format berikut, di mana *`Value`* bilangan bulat positif dan *`Unit`* dapat berupa`hour`,, `hoursday`, atau`days`:

```
rate(Value Unit)
```

Anda dapat merotasi rahasia setiap empat jam. Periode rotasi maksimum adalah 999 hari. Contoh:

- `rate(4 hours)`berarti rahasianya diputar setiap empat jam.
- `rate(1 day)`berarti rahasianya diputar setiap hari.
- `rate(10 days)`berarti rahasianya diputar setiap 10 hari.

Ekspresi Cron

Ekspresi cron Secrets Manager memiliki format berikut:

```
cron(Minutes Hours Day-of-month Month Day-of-week Year)
```

Ekspresi cron yang mencakup penambahan jam akan disetel ulang setiap hari. Misalnya, `cron(0 4/12 * * ? *)` berarti 4:00 AM, 4:00 PM, dan kemudian hari berikutnya 4:00 AM, 4:00 PM. Jadwal rotasi Secrets Manager menggunakan zona waktu UTC.

Contoh jadwal	Ekspresi
Setiap delapan jam dimulai pada tengah malam.	`cron(0 /8 * * ? *)`
Setiap delapan jam mulai pukul 8:00 pagi.	`cron(0 8/8 * * ? *)`
Setiap sepuluh jam, mulai pukul 2:00 pagi. Jendela rotasi akan dimulai pada 2:00, 12:00, dan 22:00, dan kemudian hari berikutnya pada 2:00, 12:00, dan 22:00.	`cron(0 2/10 * * ? *)`
Setiap hari pukul 10:00 pagi.	`cron(0 10 * * ? *)`
Setiap hari Sabtu pukul 18.00.	`cron(0 18 ? * SAT *)`
Hari pertama setiap bulan pukul 8:00 pagi.	`cron(0 8 1 * ? *)`
Setiap tiga bulan pada hari Minggu pertama pukul 1:00 pagi.	`cron(0 1 ? 1/3 SUN#1 *)`
Hari terakhir setiap bulan pukul 17:00.	`cron(0 17 L * ? *)`
Senin sampai Jumat pukul 8:00 pagi.	`cron(0 8 ? * MON-FRI *)`
Hari pertama dan ke-15 setiap bulan pada pukul 16:00.	`cron(0 16 1,15 * ? *)`
Minggu pertama setiap bulan pada tengah malam.	`cron(0 0 ? * SUN#1 *)`
Mulai bulan Januari, setiap 11 bulan pada hari Senin pertama di tengah malam.	`cron(0 0 ? 1/11 2#1 *)`

Persyaratan ekspresi cron di Secrets Manager

Secrets Manager memiliki beberapa batasan pada apa yang dapat Anda gunakan untuk ekspresi cron. Ekspresi cron untuk Secrets Manager harus memiliki 0 di bidang menit karena jendela rotasi Secrets Manager dimulai pada jam. Itu harus memiliki * di bidang tahun, karena Secrets Manager tidak mendukung jadwal rotasi yang terpisah lebih dari satu tahun. Tabel berikut menunjukkan opsi yang dapat Anda gunakan.

Bidang	Nilai-nilai	Wildcard
Menit	Harus 0	Tidak ada
Jam	0–23	Gunakan/(garis miring ke depan) untuk menentukan kenaikan. Misalnya `2/10` berarti setiap 10 jam dimulai pukul 2:00 pagi. Anda dapat merotasi rahasia setiap empat jam.
Day-of-month	1–31	Gunakan, (koma) untuk memasukkan nilai tambahan. Misalnya `1,15` berarti hari pertama dan ke-15 setiap bulan. Gunakan - (tanda hubung) untuk menentukan rentang. Misalnya `1-15` berarti hari 1 sampai 15 dalam sebulan. Gunakan* (tanda bintang) untuk menyertakan semua nilai di bidang. Misalnya * berarti setiap hari dalam sebulan. Wildcard ? (tanda tanya) menentukan satu atau

Bidang	Nilai-nilai	Wildcard
		yang lain. Anda tidak dapat menentukan kolom `Day-of-month` dan `Day-of-week` dalam ekspresi cron yang sama. Jika Anda menentukan sebuah nilai di salah satu kolom, maka Anda harus menggunakan ? (tanda tanya) di kolom yang lain. Gunakan/(garis miring ke depan) untuk menentukan kenaikan. Misalnya, `1/2` berarti setiap dua hari dimulai pada hari 1, dengan kata lain, hari 1, 3, 5, dan seterusnya. Gunakan L untuk menentukan hari terakhir bulan itu. Gunakan ***DAY***L untuk menentukan hari bernama terakhir bulan itu. Misalnya `SUNL` berarti hari Minggu terakhir setiap bulan.

Bidang	Nilai-nilai	Wildcard
Bulan	1—12 atau JAN—DEC	Gunakan, (koma) untuk memasukkan nilai tambahan. Misalnya, `JAN,APR,JUL,OCT` berarti Januari, April, Juli, dan Oktober. Gunakan - (tanda hubung) untuk menentukan rentang. Misalnya `1-3` berarti bulan 1 sampai 3 tahun. Gunakan* (tanda bintang) untuk menyertakan semua nilai di bidang. Misalnya `*` berarti setiap bulan. Gunakan/(garis miring ke depan) untuk menentukan kenaikan. Misalnya, `1/3` berarti setiap bulan ketiga, dimulai pada bulan 1, dengan kata lain bulan 1, 4, 7, dan 10.

Bidang	Nilai-nilai	Wildcard
Day-of-week	1—7 atau MATAHARI-SAT	Gunakan # untuk menentukan hari dalam seminggu dalam sebulan. Misalnya, `TUE#3` berarti Selasa ketiga setiap bulan. Gunakan, (koma) untuk memasukkan nilai tambahan. Misalnya `1,4` berarti hari pertama dan keempat dalam seminggu. Gunakan - (tanda hubung) untuk menentukan rentang. Misalnya `1-4` berarti hari 1 sampai 4 dalam seminggu. Gunakan* (tanda bintang) untuk menyertakan semua nilai di bidang. Misalnya * berarti setiap hari dalam seminggu. Wildcard ? (tanda tanya) menentukan satu atau yang lain. Anda tidak dapat menentukan kolom `Day-of-month` dan `Day-of-week` dalam ekspresi cron yang sama. Jika Anda menentukan sebuah nilai di salah satu kolom, maka Anda harus menggunakan ? (tanda tanya) di kolom yang lain.

Bidang	Nilai-nilai	Wildcard
		Gunakan/(garis miring ke depan) untuk menentuka n kenaikan. Misalnya, 1/2 berarti setiap hari kedua dalam seminggu, dimulai pada hari pertama, jadi hari 1, 3, 5, dan 7. Gunakan L untuk menentukan hari terakhir dalam seminggu.
Tahun	Harus *	Tidak ada

Putar sebuah AWS Secrets Manager rahasia segera

Anda hanya dapat memutar rahasia yang memiliki rotasi yang dikonfigurasi. Untuk menentukan apakah rahasia telah dikonfigurasi untuk rotasi, di konsol, lihat rahasia dan gulir ke bawah ke bagian konfigurasi Rotasi. Jika status Rotasi di aktifkan, maka rahasia dikonfigurasi untuk rotasi. Jika belum, lihat Merotasi rahasia .

Untuk memutar rahasia segera (konsol)

1. Buka konsol Secrets Manager di https://console.aws.amazon.com/secretsmanager/.
2. Pilih rahasia Anda.
3. Pada halaman detail rahasia, di bawah Konfigur asi rotasi, pilih Putar rahasia segera.
4. Di kotak dialog Put ar rahasia, pilih Putar.

AWS CLI

Example Putar rahasia segera

rotate-secretContoh berikut memulai rotasi langsung. Rahasianya harus sudah memiliki rotasi yang dikonfigurasi.

```
$ aws secretsmanager rotate-secret \
```

```
    --secret-id MyTestSecret
```

Temukan rahasia yang tidak diputar

Anda dapat menggunakan AWS Config untuk mengevaluasi rahasia Anda untuk melihat apakah mereka berputar sesuai dengan standar Anda. Anda menentukan persyaratan keamanan dan kepatuhan internal Anda untuk rahasia menggunakan AWS Config aturan. Kemudian AWS Config dapat mengidentifikasi rahasia yang tidak sesuai dengan aturan Anda. Anda juga dapat melacak perubahan metadata rahasia, konfigurasi rotasi, kunci KMS yang digunakan untuk enkripsi rahasia, fungsi rotasi Lambda, dan tag yang terkait dengan rahasia.

Jika Anda memiliki rahasia di beberapa Akun AWS dan Wilayah AWS di organisasi Anda, Anda dapat menggabungkan konfigurasi dan data kepatuhan tersebut. Untuk informasi selengkapnya, lihat agregasi Multi-account Multi-Region data.

Untuk menilai apakah rahasia berputar

1. Ikuti petunjuk tentang Mengevaluasi sumber daya Anda dengan AWS Config aturan, dan pilih dari aturan berikut:

   - `secretsmanager-rotation-enabled-check`— Memeriksa apakah rotasi dikonfigurasi untuk rahasia yang disimpan di Secrets Manager.
   - `secretsmanager-scheduled-rotation-success-check`— Memeriksa apakah rotasi sukses terakhir berada dalam frekuensi rotasi yang dikonfigurasi. Frekuensi minimum untuk cek adalah setiap hari.
   - `secretsmanager-secret-periodic-rotation`— Memeriksa apakah rahasia diputar dalam jumlah hari yang ditentukan.
2. Secara opsional, konfigurasikan AWS Config untuk memberi tahu Anda saat rahasia tidak sesuai. Untuk informasi selengkapnya, lihat Pemberitahuan yang AWS Config mengirim ke topik Amazon SNS.

Batalkan rotasi otomatis di Secrets Manager

Jika Anda mengonfigurasi rotasi otomatis untuk rahasia dan Anda ingin berhenti memutarnya, Anda dapat membatalkan rotasi.

Untuk membatalkan rotasi otomatis

1. Buka konsol Secrets Manager di https://console.aws.amazon.com/secretsmanager/.
2. Pilih rahasiamu.
3. Pada halaman detail rahasia, di bawah konfigurasi Rotasi, pilih Edit rotasi.
4. Dalam kotak dialog Edit konfigurasi rotasi, matikan Rotasi otomatis, lalu pilih Simpan.

   Secrets Manager menyimpan informasi konfigurasi rotasi sehingga Anda dapat menggunakannya di masa depan jika Anda memutuskan untuk mengaktifkan kembali rotasi.

AWS Secrets Manager rahasia yang dikelola oleh AWS layanan lain

Banyak AWS layanan menyimpan dan menggunakan rahasia di AWS Secrets Manager. Dalam beberapa kasus, rahasia ini adalah rahasia yang dikelola, yang berarti bahwa layanan yang membuatnya membantu mengelolanya. Misalnya, beberapa rahasia terkelola menyertakan rotasi terkelola, jadi Anda tidak perlu mengonfigurasi rotasi sendiri. Layanan pengelolaan mungkin juga membatasi Anda untuk memperbarui rahasia atau menghapusnya tanpa periode pemulihan, yang membantu mencegah pemadaman karena layanan pengelolaan bergantung pada rahasianya.

> **Note**
>
> Rahasia terkelola hanya dapat dibuat oleh AWS layanan yang mengelolanya.

Rahasia terkelola menggunakan konvensi penamaan yang menyertakan ID layanan pengelolaan untuk membantu mengidentifikasinya.

```
Secret name: ServiceID!MySecret
Secret ARN : arn:aws:us-east-1:ServiceID!MySecret-a1b2c3
```

ID untuk layanan yang mengelola rahasia

- `appflow` – the section called “Amazon AppFlow”
- `databrew` – the section called “AWS Glue DataBrew”
- `datasync` – the section called “AWS DataSync”
- `directconnect` – the section called “Direct Connect”
- `ecs-sc` – the section called “Amazon Elastic Container Service”
- `events` – the section called “Amazon EventBridge”
- `marketplace-deployment` – the section called “AWS Marketplace”
- `opsworks-cm` – the section called “AWS OpsWorks for Chef Automate”
- `pcs` – the section called “AWS Layanan Komputasi Paralel”
- `rds` – the section called “Amazon RDS”
- `redshift` – the section called “Amazon Redshift”

- `sqlworkbench` – the section called “Editor kueri Amazon Redshift v2”

Untuk menemukan rahasia yang dikelola oleh AWS layanan lain, lihat Menemukan rahasia yang dikelola.

Layanan AWS yang menggunakan AWS Secrets Manager rahasia

Dapatkan informasi tentang bagaimana masing-masing hal berikut Layanan AWS berintegrasi dengan Secrets Manager.

- Bagaimana AWS App Runner menggunakan AWS Secrets Manager
- Bagaimana AWS Penggunaan App2Container AWS Secrets Manager
- Bagaimana AWS AppConfig menggunakan AWS Secrets Manager
- Bagaimana Amazon AppFlow menggunakan AWS Secrets Manager
- Bagaimana AWS AppSync menggunakan AWS Secrets Manager
- Bagaimana Amazon Athena menggunakan AWS Secrets Manager
- Bagaimana Amazon Aurora menggunakan AWS Secrets Manager
- Bagaimana AWS CodeBuild menggunakan AWS Secrets Manager
- Bagaimana Amazon Data Firehose menggunakan AWS Secrets Manager
- Bagaimana AWS DataSync menggunakan AWS Secrets Manager
- Bagaimana Amazon DataZone menggunakan AWS Secrets Manager
- Bagaimana AWS Direct Connect menggunakan AWS Secrets Manager
- Bagaimana AWS Directory Service menggunakan AWS Secrets Manager
- Bagaimana Amazon DocumentDB (dengan kompatibilitas MongoDB) menggunakan AWS Secrets Manager
- Bagaimana AWS Elastic Beanstalk menggunakan AWS Secrets Manager
- Bagaimana Amazon Elastic Container Registry menggunakan AWS Secrets Manager
- Amazon Elastic Container Service
- Bagaimana Amazon ElastiCache menggunakan AWS Secrets Manager
- Bagaimana AWS Elemental Live menggunakan AWS Secrets Manager
- Bagaimana AWS Elemental MediaConnect menggunakan AWS Secrets Manager
- Bagaimana AWS Elemental MediaConvert menggunakan AWS Secrets Manager
- Bagaimana AWS Elemental MediaLive menggunakan AWS Secrets Manager

- Bagaimana AWS Elemental MediaPackage menggunakan AWS Secrets Manager
- Bagaimana AWS Elemental MediaTailor menggunakan AWS Secrets Manager
- Bagaimana Amazon EMR menggunakan Secrets Manager
- Bagaimana Amazon EventBridge menggunakan AWS Secrets Manager
- Bagaimana Amazon FSx menggunakan AWS Secrets Manager rahasia
- Bagaimana AWS Glue DataBrew menggunakan AWS Secrets Manager
- Bagaimana AWS Glue Studio menggunakan AWS Secrets Manager
- Bagaimana AWS IoT SiteWise menggunakan AWS Secrets Manager
- Bagaimana Amazon Kendra menggunakan AWS Secrets Manager
- Bagaimana Amazon Kinesis Video Streams menggunakan AWS Secrets Manager
- Bagaimana AWS Launch Wizard menggunakan AWS Secrets Manager
- Cara Amazon Lookout for Metrics menggunakan AWS Secrets Manager
- Bagaimana Amazon Managed Grafana menggunakan AWS Secrets Manager
- Bagaimana AWS Managed Services menggunakan AWS Secrets Manager
- Bagaimana Amazon Managed Streaming for Apache Kafka menggunakan AWS Secrets Manager
- Bagaimana Amazon Managed Workflow untuk Apache Airflow menggunakan AWS Secrets Manager
- AWS Marketplace
- Bagaimana AWS Migration Hub menggunakan AWS Secrets Manager
- Bagaimana AWS Penggunaan Layanan Komputasi Paralel AWS Secrets Manager
- Bagaimana AWS ParallelCluster menggunakan AWS Secrets Manager
- Bagaimana Amazon Q menggunakan Secrets Manager
- Bagaimana Amazon OpenSearch Tertelan menggunakan Secrets Manager
- Bagaimana AWS OpsWorks for Chef Automate menggunakan AWS Secrets Manager
- Bagaimana Amazon Quick menggunakan AWS Secrets Manager
- Bagaimana Amazon RDS menggunakan AWS Secrets Manager
- Bagaimana Amazon Redshift menggunakan AWS Secrets Manager
- Editor kueri Amazon Redshift v2
- Bagaimana Amazon SageMaker AI menggunakan AWS Secrets Manager
- Bagaimana AWS Schema Conversion Tool menggunakan AWS Secrets Manager

- Bagaimana Amazon Timestream untuk InfluxDB menggunakan AWS Secrets Manager
- Bagaimana AWS Toolkit for JetBrains menggunakan AWS Secrets Manager
- Bagaimana AWS Transfer Family menggunakan AWS Secrets Manager rahasia
- Bagaimana AWS Wickr menggunakan AWS Secrets Manager rahasia

Bagaimana AWS App Runner menggunakan AWS Secrets Manager

AWS App Runner adalah AWS layanan yang menyediakan cara cepat, sederhana, dan hemat biaya untuk menyebarkan dari kode sumber atau gambar kontainer langsung ke aplikasi web yang dapat diskalakan dan aman di Cloud. AWS Anda tidak perlu mempelajari teknologi baru, memutuskan layanan komputasi mana yang akan digunakan, atau mengetahui cara menyediakan dan mengonfigurasi AWS sumber daya.

Dengan App Runner, Anda dapat mereferensikan rahasia dan konfigurasi sebagai variabel lingkungan dalam layanan Anda saat membuat layanan atau memperbarui konfigurasi layanan. Untuk informasi selengkapnya, lihat Mereferensikan variabel lingkungan dan Mengelola variabel lingkungan di Panduan AWS App Runner Pengembang.

Bagaimana AWS Penggunaan App2Container AWS Secrets Manager

AWS App2Container adalah alat baris perintah untuk membantu Anda mengangkat dan menggeser aplikasi yang berjalan di pusat data lokal atau di mesin virtual, sehingga mereka berjalan dalam wadah yang dikelola oleh Amazon ECS, Amazon EKS, atau. AWS App Runner

App2Container menggunakan Secrets Manager untuk mengelola kredensyal untuk menghubungkan mesin pekerja Anda ke server aplikasi untuk menjalankan perintah jarak jauh. Untuk informasi selengkapnya, lihat Mengelola rahasia untuk AWS App2Container di Panduan Pengguna AWS App2Container.

Bagaimana AWS AppConfig menggunakan AWS Secrets Manager

AWS AppConfig adalah kemampuan AWS Systems Manager yang dapat Anda gunakan untuk membuat, mengelola, dan menyebarkan konfigurasi aplikasi dengan cepat. Konfigurasi dapat berisi data kredensi atau informasi sensitif lainnya yang disimpan di Secrets Manager. Saat membuat profil konfigurasi bentuk bebas, Anda dapat memilih Secrets Manager sebagai sumber data konfigurasi Anda. Untuk informasi selengkapnya, lihat Membuat profil konfigurasi bentuk bebas di Panduan AWS AppConfig Pengguna. Untuk informasi tentang cara AWS AppConfig menangani rahasia

yang mengaktifkan rotasi otomatis, lihat Rotasi kunci Secrets Manager di Panduan AWS AppConfig Pengguna.

Bagaimana Amazon AppFlow menggunakan AWS Secrets Manager

Amazon AppFlow adalah layanan integrasi yang dikelola sepenuhnya yang memungkinkan Anda bertukar data dengan aman antara aplikasi perangkat lunak sebagai layanan (SaaS), seperti Salesforce, dan, seperti Amazon Simple Storage Service (Amazon S3) dan Amazon Redshift. Layanan AWS

Di Amazon AppFlow, saat Anda mengonfigurasi aplikasi SaaS sebagai sumber atau tujuan, Anda membuat koneksi. Ini termasuk informasi yang diperlukan untuk menghubungkan ke aplikasi SaaS, seperti token otentikasi, nama pengguna, dan kata sandi. Amazon AppFlow menyimpan data koneksi Anda dalam rahasia yang dikelola Secrets Manager dengan awalan`appflow`. Biaya penyimpanan rahasia sudah termasuk dengan biaya untuk Amazon AppFlow. Untuk informasi selengkapnya, lihat Perlindungan data AppFlow di Amazon di Panduan AppFlow Pengguna Amazon.

Bagaimana AWS AppSync menggunakan AWS Secrets Manager

AWS AppSync menyediakan antarmuka GraphQL yang kuat dan dapat diskalakan bagi pengembang aplikasi untuk menggabungkan data dari berbagai sumber, termasuk Amazon DynamoDB,, dan API HTTP. AWS Lambda

AWS AppSync menggunakan kredensi dalam rahasia Secrets Manager untuk terhubung ke Amazon RDS dan Aurora. Untuk informasi lebih lanjut, lihat Tutorial: Aurora Tanpa Server di Panduan Pengembang.AWS AppSync

Bagaimana Amazon Athena menggunakan AWS Secrets Manager

Amazon Athena adalah layanan kueri interaktif yang memudahkan untuk menganalisis data di Amazon Simple Storage Service (Amazon S3) menggunakan SQL standar.

Konektor sumber data Amazon Athena dapat menggunakan fitur Kueri Federasi Athena dengan rahasia Secrets Manager untuk menanyakan data. Untuk informasi selengkapnya, lihat Menggunakan Kueri Federasi Amazon Athena di Panduan Pengguna Amazon Athena.

Bagaimana Amazon Aurora menggunakan AWS Secrets Manager

Amazon Aurora adalah mesin database relasional yang dikelola sepenuhnya yang kompatibel dengan MySQL dan PostgreSQL.

Untuk mengelola kredensi pengguna master untuk Aurora, Aurora dapat membuat rahasia terkelola untuk Anda. Anda dituntut untuk rahasia itu. Aurora juga mengelola rotasi untuk kredensil ini. Untuk informasi selengkapnya, lihat Manajemen kata sandi dengan Amazon Aurora dan AWS Secrets Manager di Panduan Pengguna Amazon Aurora.

Untuk kredensi Aurora lainnya, lihat. Buat rahasia

Saat Anda memanggil Amazon RDS Data API, Anda dapat meneruskan kredensil untuk database dengan menggunakan rahasia di Secrets Manager. Untuk informasi lebih lanjut, lihat Menggunakan API Data untuk Aurora Nirserver di Panduan Pengguna Amazon Aurora.

Saat Anda menggunakan editor kueri Amazon RDS untuk menyambung ke database, Anda dapat menyimpan kredensyal untuk database di Secrets Manager. Untuk informasi selengkapnya, lihat Menggunakan editor kueri di Panduan Pengguna Amazon RDS.

Bagaimana AWS CodeBuild menggunakan AWS Secrets Manager

AWS CodeBuild adalah layanan build yang dikelola sepenuhnya di cloud. CodeBuild mengkompilasi kode sumber Anda, menjalankan pengujian unit, dan menghasilkan artefak yang siap digunakan.

Anda dapat menyimpan kredensi registri pribadi Anda menggunakan Secrets Manager. Untuk informasi selengkapnya, lihat Registri pribadi dengan AWS Secrets Manager contoh untuk CodeBuild di Panduan AWS CodeBuild Pengguna.

Bagaimana Amazon Data Firehose menggunakan AWS Secrets Manager

Anda dapat menggunakan Amazon Data Firehose untuk mengirimkan data streaming real-time ke berbagai tujuan streaming. Ketika tujuan memerlukan kredensyal atau kunci, Firehose mengambil rahasia dari Secrets Manager saat runtime untuk terhubung ke tujuan. Untuk informasi selengkapnya, lihat Mengautentikasi dengan Firehose Data Amazon AWS Secrets Manager di Panduan Pengembang Firehose Data Amazon.

Bagaimana AWS DataSync menggunakan AWS Secrets Manager

AWS DataSync adalah layanan transfer data online yang menyederhanakan, mengotomatisasi, dan mempercepat pemindahan data antara sistem penyimpanan dan layanan.

Beberapa sistem penyimpanan yang didukung oleh DataSync memerlukan kredensi untuk membaca dan menulis data. DataSync menggunakan Secrets Manager untuk menyimpan atau mengakses kredensi penyimpanan. Anda dapat mengonfigurasi DataSync untuk membuat rahasia atas nama

Anda atau Anda dapat memberikan rahasia khusus. Service-managed rahasia dimulai dengan awalan`aws-datasync`. Anda hanya dikenakan biaya untuk penggunaan rahasia yang Anda buat di luar DataSync. Lihat Menyediakan kredensil untuk lokasi penyimpanan di AWS DataSync Panduan Pengguna.

Bagaimana Amazon DataZone menggunakan AWS Secrets Manager

Amazon DataZone adalah layanan manajemen data yang memungkinkan Anda membuat katalog, menemukan, mengatur, berbagi, dan menganalisis data Anda. Anda dapat menggunakan aset data dari tabel dan tampilan dari klaster Amazon Redshift yang dirayapi menggunakan pekerjaan. Perayap AWS Glue Untuk terhubung ke Amazon Redshift, Anda memberikan DataZone kredensi Amazon dalam rahasia Secrets Manager. Untuk informasi selengkapnya, lihat Membuat sumber data untuk database Amazon Redshift menggunakan AWS Glue koneksi baru di DataZone Panduan Pengguna Amazon.

Bagaimana AWS Direct Connect menggunakan AWS Secrets Manager

Direct Connect menghubungkan jaringan internal Anda ke Direct Connect lokasi melalui kabel serat optik Ethernet standar. Dengan koneksi ini, Anda dapat membuat antarmuka virtual langsung ke publik Layanan AWS.

Direct Connect menyimpan nama kunci asosiasi konektivitas dan konektivitas asosiasi key CKN/CAK pair (pair) dalam rahasia terkelola dengan awalan`directconnect`. Biaya rahasia sudah termasuk dengan biaya untuk Direct Connect. Untuk memperbarui rahasia, Anda harus menggunakan Direct Connect bukan Secrets Manager. Untuk informasi selengkapnya, lihat Mengaitkan MacSec CKN/CAK dengan LAG di Panduan Direct Connect Pengguna.

Bagaimana AWS Directory Service menggunakan AWS Secrets Manager

Directory Service menyediakan beberapa cara untuk menggunakan Microsoft Active Directory (AD) dengan AWS layanan lain. Anda dapat bergabung dengan instans Amazon EC2 ke direktori Anda menggunakan rahasia untuk kredensi. Untuk informasi selengkapnya, di Panduan Direct Connect Pengguna, lihat:

- Bergabunglah dengan instans Linux EC2 dengan mulus ke direktori AWS Microsoft AD yang Dikelola
- Bergabunglah dengan instans EC2 Linux dengan mulus ke direktori AD Connector Anda
- Bergabunglah dengan instans Linux EC2 dengan mulus ke direktori Simple AD Anda

Bagaimana Amazon DocumentDB (dengan kompatibilitas MongoDB) menggunakan AWS Secrets Manager

Amazon DocumentDB (dengan kompatibilitas MongoDB) adalah layanan database dokumen yang dikelola sepenuhnya yang mendukung beban kerja MongoDB. Amazon DocumentDB terintegrasi dengan Secrets Manager untuk mengelola kata sandi pengguna utama untuk cluster Anda, meningkatkan keamanan, dan menyederhanakan manajemen kredensi.

Amazon DocumentDB menghasilkan kata sandi, menyimpannya di Secrets Manager, dan mengelola pengaturan rahasia. Secara default, Amazon DocumentDB memutar rahasia setiap tujuh hari, tetapi Anda dapat mengubah jadwal rotasi jika diperlukan. Saat membuat atau memodifikasi klaster Amazon DocumentDB, Anda dapat menentukan bahwa klaster tersebut harus mengelola kata sandi pengguna utama di Secrets Manager. Untuk informasi selengkapnya, lihat Manajemen kata sandi dengan Amazon DocumentDB dan Secrets Manager di Panduan Pengembang Amazon DocumentDB.

Bagaimana AWS Elastic Beanstalk menggunakan AWS Secrets Manager

Dengan AWS Elastic Beanstalk, Anda dapat dengan cepat menyebarkan dan mengelola aplikasi di AWS Cloud tanpa harus belajar tentang infrastruktur yang menjalankan aplikasi tersebut. Elastic Beanstalk dapat meluncurkan lingkungan Docker dengan membuat gambar yang dijelaskan dalam Dockerfile atau menarik gambar Docker jarak jauh. Untuk mengautentikasi dengan registri online yang menampung repositori pribadi, Elastic Beanstalk menggunakan rahasia Secrets Manager. Untuk informasi selengkapnya, lihat konfigurasi Docker di Panduan AWS Elastic Beanstalk Pengembang.

Bagaimana Amazon Elastic Container Registry menggunakan AWS Secrets Manager

Amazon Elastic Container Registry (Amazon ECR) adalah AWS layanan registri gambar kontainer terkelola yang aman, terukur, dan andal. Anda dapat menggunakan CLI Docker, atau klien pilihan Anda, untuk mendorong dan menarik gambar ke dan dari repositori Anda. Untuk setiap registri upstream yang berisi gambar yang ingin Anda cache di registri pribadi Amazon ECR Anda, Anda harus membuat aturan cache pull through. Untuk registrasi upstream yang memerlukan otentikasi, Anda harus menyimpan kredensialnya dalam rahasia Secrets Manager. Anda dapat membuat rahasia Secrets Manager di konsol Amazon ECR atau Secrets Manager. Untuk informasi selengkapnya, lihat Membuat aturan cache tarik melalui di Panduan Pengguna Amazon ECR.

Amazon Elastic Container Service

Amazon Elastic Container Service (Amazon ECS) adalah layanan orkestrasi kontainer terkelola penuh yang membantu Anda menerapkan, mengelola, dan menskalakan aplikasi kontainer dengan mudah. Anda dapat menyuntikkan data sensitif ke dalam kontainer dengan mereferensikan rahasia Secrets Manager. Untuk informasi selengkapnya, lihat halaman berikut di Panduan Pengembang Layanan Amazon Elastic Container:

- Tutorial: Menentukan data sensitif menggunakan rahasia Secrets Manager
- Ambil rahasia secara terprogram melalui aplikasi Anda
- Ambil rahasia melalui variabel lingkungan
- Ambil rahasia untuk konfigurasi logging

Amazon ECS mendukung volume FSx for Windows File Server untuk kontainer. Amazon ECS menggunakan kredensil yang disimpan dalam rahasia Secrets Manager untuk domain bergabung dengan Active Directory dan melampirkan sistem file FSx for Windows File Server. Untuk informasi selengkapnya, lihat Tutorial: Menggunakan sistem file FSx for Windows File Server dengan Amazon ECS dan FSx for Windows File Server volume di Amazon Elastic Container Service Developer Guide.

Anda dapat mereferensikan gambar kontainer di registri pribadi di luar AWS yang memerlukan otentikasi dengan menggunakan rahasia Secrets Manager dengan kredenal registri. Untuk informasi selengkapnya, lihat Autentikasi registri pribadi untuk tugas di Panduan Pengembang Layanan Amazon Elastic Container.

Saat Anda menggunakan Amazon ECS Service Connect, Amazon ECS menggunakan rahasia yang dikelola Secrets Manager untuk menyimpan sertifikat AWS Private Certificate Authority TLS. Biaya penyimpanan rahasia sudah termasuk dengan biaya untuk Amazon ECS. Untuk memperbarui rahasia, Anda harus menggunakan Amazon ECS daripada Secrets Manager. Untuk informasi selengkapnya, lihat TLS dengan Service Connect di Panduan Pengembang Layanan Amazon Elastic Container.

Bagaimana Amazon ElastiCache menggunakan AWS Secrets Manager

Di dalam ElastiCache Anda dapat menggunakan fitur yang disebut Role-Based Access Control (RBAC) untuk mengamankan cluster. Anda dapat menyimpan kredensil ini di Secrets Manager. Secrets Manager menyediakan template rotasi untuk jenis rahasia ini. Untuk informasi selengkapnya, lihat Memutar kata sandi secara otomatis untuk pengguna di Panduan ElastiCache Pengguna Amazon.

Bagaimana AWS Elemental Live menggunakan AWS Secrets Manager

AWS Elemental Live adalah layanan video real-time yang memungkinkan Anda membuat output langsung untuk siaran dan pengiriman streaming.

AWS Elemental Live menggunakan ARN rahasia untuk mendapatkan rahasia yang berisi kunci enkripsi dari Secrets Manager. Elemental Live menggunakan kunci enkripsi encrypt/decrypt ke video. Untuk informasi selengkapnya, lihat Cara pengiriman dari AWS Elemental Live ke MediaConnect bekerja saat runtime di Panduan Pengguna Elemental Live.

Bagaimana AWS Elemental MediaConnect menggunakan AWS Secrets Manager

AWS Elemental MediaConnect adalah layanan yang memudahkan penyiar dan penyedia video premium lainnya untuk secara andal menyerap video langsung ke dalam AWS Cloud dan mendistribusikannya ke beberapa tujuan di dalam atau di luar. AWS Cloud

Anda dapat menggunakan enkripsi kunci statis untuk melindungi sumber, output, dan hak Anda, dan Anda menyimpan kunci enkripsi Anda. AWS Secrets Manager Untuk informasi selengkapnya, lihat Enkripsi kunci statis AWS Elemental MediaConnect di Panduan AWS Elemental MediaConnect Pengguna.

Bagaimana AWS Elemental MediaConvert menggunakan AWS Secrets Manager

AWS Elemental MediaConvert adalah layanan pemrosesan video berbasis file yang menyediakan pemrosesan video yang dapat diskalakan untuk pemilik konten dan distributor dengan pustaka media dalam berbagai ukuran. Untuk digunakan MediaConvert untuk menyandikan tanda air Kantar, Anda menggunakan Secrets Manager untuk menyimpan kredenal Kantar Anda. Untuk informasi selengkapnya, lihat Menggunakan Kantar untuk watermarking audio dalam AWS Elemental MediaConvert output di Panduan Pengguna.AWS Elemental MediaConvert

Bagaimana AWS Elemental MediaLive menggunakan AWS Secrets Manager

AWS Elemental MediaLive adalah layanan video real-time yang memungkinkan Anda membuat output langsung untuk siaran dan pengiriman streaming. Jika organisasi Anda menggunakan AWS Elemental Link perangkat dengan AWS Elemental MediaLive atau AWS Elemental MediaConnect,

Anda harus menggunakan perangkat dan mengonfigurasi perangkat. Untuk informasi selengkapnya, lihat Menyiapkan MediaLive sebagai entitas tepercaya di Panduan MediaLive Pengguna.

Bagaimana AWS Elemental MediaPackage menggunakan AWS Secrets Manager

AWS Elemental MediaPackage adalah layanan pengemasan dan originasi video tepat waktu yang berjalan di. AWS Cloud Dengan MediaPackage, Anda dapat memberikan streaming video yang sangat aman, terukur, dan andal ke berbagai perangkat pemutaran dan jaringan pengiriman konten (CDN). Untuk informasi selengkapnya, lihat Akses Secrets Manager untuk otorisasi CDN di AWS Elemental MediaPackage Panduan Pengguna.

Bagaimana AWS Elemental MediaTailor menggunakan AWS Secrets Manager

AWS Elemental MediaTailor adalah layanan penyisipan iklan dan perakitan saluran yang dapat diskalakan yang berjalan di. AWS Cloud

MediaTailor mendukung otentikasi token akses Secrets Manager ke lokasi sumber Anda. Dengan otentikasi token akses Secrets Manager, MediaTailor gunakan rahasia Secrets Manager untuk mengautentikasi permintaan ke asal Anda. Untuk informasi selengkapnya, lihat Mengonfigurasi otentikasi token AWS Secrets Manager akses di AWS Elemental MediaTailor Panduan Pengguna.

Bagaimana Amazon EMR menggunakan Secrets Manager

Amazon EMR adalah platform yang menyederhanakan menjalankan kerangka kerja data besar, seperti Apache Hadoop dan Apache Spark, untuk memproses dan menganalisis sejumlah besar data AWS . Saat Anda menggunakan kerangka kerja ini dan proyek sumber terbuka terkait seperti Apache Hive dan Apache Pig, Anda dapat memproses data untuk analitik dan beban kerja intelijen bisnis. Anda juga dapat menggunakan Amazon EMR untuk mengubah dan memindahkan sejumlah besar data ke dalam dan keluar dari penyimpanan AWS data dan database lainnya, seperti Amazon S3 dan Amazon DynamoDB.

Bagaimana Amazon EMR berjalan di Amazon EC2 menggunakan Secrets Manager

Saat membuat cluster di Amazon EMR, Anda dapat memberikan data konfigurasi aplikasi ke cluster dengan rahasia di Secrets Manager. Untuk informasi selengkapnya, lihat Menyimpan data konfigurasi sensitif di Secrets Manager di Panduan Manajemen EMR Amazon.

Selain itu, saat membuat Notebook EMR, Anda dapat menyimpan kredensyal Git-based registri pribadi menggunakan Secrets Manager. Untuk informasi selengkapnya, lihat Menambahkan Git-based Repositori ke Amazon EMR di Panduan Manajemen EMR Amazon.

Bagaimana EMR Serverless menggunakan Secrets Manager

EMR Serverless menyediakan lingkungan runtime tanpa server untuk menyederhanakan pengoperasian aplikasi analitik sehingga Anda tidak perlu mengonfigurasi, mengoptimalkan, mengamankan, atau mengoperasikan cluster.

Anda dapat menyimpan data Anda AWS Secrets Manager dan kemudian menggunakan ID rahasia dalam konfigurasi EMR Tanpa Server Anda. Dengan cara ini, Anda tidak meneruskan data konfigurasi sensitif dalam teks biasa dan mengeksposnya ke API eksternal.

Untuk informasi selengkapnya, lihat Secrets Manager untuk perlindungan data dengan EMR Tanpa Server di Panduan Pengguna Tanpa Server Amazon EMR.

Bagaimana Amazon EventBridge menggunakan AWS Secrets Manager

Amazon EventBridge adalah layanan bus acara tanpa server yang dapat Anda gunakan untuk menghubungkan aplikasi Anda dengan data dari berbagai sumber.

Saat Anda membuat tujuan Amazon EventBridge API, EventBridge menyimpan koneksi untuk itu dalam rahasia yang dikelola Secrets Manager dengan awalan`events`. Biaya penyimpanan rahasia sudah termasuk dengan biaya untuk menggunakan tujuan API. Untuk memperbarui rahasia, Anda harus menggunakan EventBridge bukan Secrets Manager. Untuk informasi selengkapnya, lihat tujuan API di Panduan EventBridge Pengguna Amazon.

Bagaimana Amazon FSx menggunakan AWS Secrets Manager rahasia

Amazon FSx for Windows File Server menyediakan server file Microsoft Windows yang terkelola penuh, yang didukung oleh sistem file Windows sepenuhnya asli. Saat Anda membuat atau mengelola pembagian file, Anda dapat meneruskan kredensi dari rahasia. AWS Secrets Manager Untuk informasi selengkapnya, lihat Berbagi file dan Memigrasi konfigurasi berbagi file ke Amazon FSx di Panduan Pengguna Amazon FSx for Windows File Server.

Bagaimana AWS Glue DataBrew menggunakan AWS Secrets Manager

AWS Glue DataBrew adalah alat persiapan data visual yang dapat Anda gunakan untuk membersihkan dan menormalkan data tanpa menulis kode apa pun. Pada tahun DataBrew,

satu set langkah transformasi data disebut resep. AWS Glue DataBrew menyediakan `DETERMINISTIC_DECRYPT`, `DETERMINISTIC_ENCRYPT`, dan langkah-langkah `CRYPTOGRAPHIC_HASH`resep untuk melakukan transformasi pada informasi yang dapat diidentifikasi secara pribadi (PII) dalam kumpulan data, yang menggunakan kunci enkripsi yang disimpan dalam rahasia Secrets Manager. Jika Anda menggunakan rahasia DataBrew default untuk menyimpan kunci enkripsi, DataBrew buat rahasia terkelola dengan awalan`databrew`. Biaya penyimpanan rahasia sudah termasuk dengan biaya untuk menggunakan DataBrew. Jika Anda membuat rahasia baru untuk menyimpan kunci enkripsi, DataBrew buat rahasia dengan awalan`AwsGlueDataBrew`. Anda dituntut untuk rahasia itu.

Bagaimana AWS Glue Studio menggunakan AWS Secrets Manager

AWS Glue Studio adalah antarmuka grafis yang memudahkan untuk membuat, menjalankan, dan memantau pekerjaan ekstrak, transformasi, dan memuat (ETL) di AWS Glue. Anda dapat menggunakan Amazon OpenSearch Service sebagai penyimpanan data untuk pekerjaan ekstrak, transformasi, dan pemuatan (ETL) Anda dengan mengonfigurasi Elasticsearch Spark Connector di. AWS Glue Studio Untuk terhubung ke OpenSearch cluster, Anda dapat menggunakan rahasia di Secrets Manager. Untuk informasi lebih lanjut, lihat Tutorial: Menggunakan Konektor AWS Glue untuk Elasticsearch di Panduan AWS Glue Pengembang.

Bagaimana AWS IoT SiteWise menggunakan AWS Secrets Manager

AWS IoT SiteWise adalah layanan terkelola yang memungkinkan Anda mengumpulkan, memodelkan, menganalisis, dan memvisualisasikan data dari peralatan industri dalam skala besar. Anda dapat menggunakan AWS IoT SiteWise konsol untuk membuat gateway. Kemudian tambahkan sumber data, server lokal atau peralatan industri yang terhubung ke gateway. Jika sumber Anda memerlukan otentikasi, gunakan rahasia untuk mengautentikasi. Untuk informasi selengkapnya, lihat Mengonfigurasi autentikasi sumber data di AWS IoT SiteWise Panduan Pengguna.

Bagaimana Amazon Kendra menggunakan AWS Secrets Manager

Amazon Kendra adalah layanan pencarian yang sangat akurat dan cerdas yang memungkinkan pengguna Anda untuk mencari data yang tidak terstruktur dan terstruktur menggunakan pemrosesan bahasa alami dan algoritme pencarian lanjutan.

Anda dapat mengindeks dokumen yang disimpan dalam database dengan menentukan rahasia yang berisi kredensil untuk database. Untuk informasi selengkapnya, lihat Menggunakan sumber data database di Panduan Pengguna Amazon Kendra.

Bagaimana Amazon Kinesis Video Streams menggunakan AWS Secrets Manager

Anda dapat menggunakan Amazon Kinesis Video Streams untuk terhubung ke kamera IP di tempat pelanggan, merekam dan menyimpan video secara lokal dari kamera, dan mengalirkan video ke cloud untuk penyimpanan jangka panjang, pemutaran, dan pemrosesan analitis. Untuk merekam dan mengunggah media dari kamera IP, Anda menggunakan Agen Edge Streams Kinesis Video Streams ke. AWS IoT Greengrass Anda menyimpan kredensyal yang diperlukan untuk mengakses file media yang dialirkan ke kamera dalam rahasia Secrets Manager. Untuk informasi selengkapnya, lihat Menerapkan Agen Edge Amazon Kinesis Video Streams AWS IoT Greengrass ke dalam Panduan Pengembang Amazon Kinesis Video Streams.

Bagaimana AWS Launch Wizard menggunakan AWS Secrets Manager

AWS Launch Wizard untuk Active Directory adalah layanan yang menerapkan praktik terbaik AWS Cloud aplikasi untuk memandu Anda melalui pengaturan infrastruktur Direktori Aktif baru, atau menambahkan pengontrol domain ke infrastruktur yang ada, baik di dalam AWS Cloud maupun di tempat.

AWS Launch Wizard memerlukan kredensi administrator domain untuk ditambahkan ke Secrets Manager untuk bergabung dengan pengontrol domain Anda ke Active Directory. Untuk informasi selengkapnya, lihat Mengatur AWS Launch Wizard untuk Active Directory di Panduan AWS Launch Wizard Pengguna.

Cara Amazon Lookout for Metrics menggunakan AWS Secrets Manager

Amazon Lookout for Metrics adalah layanan yang menemukan anomali dalam data Anda, menentukan akar penyebabnya, dan memungkinkan Anda mengambil tindakan dengan cepat. Anda dapat menggunakan Amazon Redshift atau Amazon RDS sebagai sumber data untuk detektor Lookout for Metrics. Untuk mengkonfigurasi sumber data, Anda menggunakan rahasia yang berisi kata sandi database. Untuk informasi selengkapnya, lihat Menggunakan Amazon RDS dengan Lookout for Metrics dan Menggunakan Amazon Redshift dengan Lookout for Metrics di Panduan Pengembang Lookout for Metrics.

Bagaimana Amazon Managed Grafana menggunakan AWS Secrets Manager

Grafana Terkelola Amazon adalah layanan visualisasi data yang dikelola sepenuhnya dan aman yang dapat Anda gunakan untuk menanyakan, mengkorelasikan, dan memvisualisasikan metrik, log, dan jejak operasional secara instan dari berbagai sumber. Saat Anda menggunakan Amazon Redshift sebagai sumber data, Anda dapat memberikan kredensi Amazon Redshift dengan menggunakan rahasia. AWS Secrets Manager Untuk informasi selengkapnya, lihat Mengonfigurasi Amazon Redshift di Panduan Pengguna Grafana Terkelola Amazon.

Bagaimana AWS Managed Services menggunakan AWS Secrets Manager

AWS Managed Services adalah layanan perusahaan yang menyediakan manajemen AWS infrastruktur Anda yang berkelanjutan. Mode AMS Self-Service Provisioning (SSP) menyediakan akses penuh ke kemampuan native Layanan AWS dan API di akun terkelola AMS. Untuk informasi tentang cara meminta akses ke Secrets Manager di AMS, lihat AWS Secrets Manager (Penyediaan layanan mandiri AMS) di Panduan Pengguna Lanjutan AMS.

Bagaimana Amazon Managed Streaming for Apache Kafka menggunakan AWS Secrets Manager

Amazon Managed Streaming for Apache Kafka (Amazon MSK) adalah layanan yang dikelola sepenuhnya yang memungkinkan Anda membangun dan menjalankan aplikasi yang menggunakan Apache Kafka untuk memproses data streaming. Anda dapat mengontrol akses ke kluster MSK Amazon Anda menggunakan nama pengguna dan kata sandi yang disimpan dan diamankan menggunakan. AWS Secrets Manager Untuk informasi selengkapnya, lihat Autentikasi nama pengguna dan kata sandi dengan AWS Secrets Manager di Panduan Developer Amazon Managed Streaming for Apache Kafka.

Bagaimana Amazon Managed Workflow untuk Apache Airflow menggunakan AWS Secrets Manager

Alur Kerja Terkelola Amazon untuk Apache Airflow adalah layanan orkestrasi terkelola untuk Apache Airflow yang memudahkan penyiapan dan pengoperasian jaringan data ujung ke ujung di cloud dalam skala besar.

Anda dapat mengonfigurasi koneksi Apache Airflow menggunakan rahasia Secrets Manager. Untuk informasi selengkapnya, lihat Mengonfigurasi koneksi Apache Airflow menggunakan rahasia Secrets

Manager dan Menggunakan kunci AWS Secrets Manager rahasia untuk variabel Apache Airflow di Amazon Managed Workflow for Apache Airflow User Guide.

AWS Marketplace

Ketika Anda menggunakan AWS Marketplace Quick Launch, AWS Marketplace mendistribusikan perangkat lunak Anda bersama dengan kunci lisensi. AWS Marketplace menyimpan kunci lisensi di akun Anda sebagai rahasia yang dikelola Secrets Manager. Biaya penyimpanan rahasia sudah termasuk dengan biaya untuk AWS Marketplace. Untuk memperbarui rahasia, Anda harus menggunakan AWS Marketplace bukan Secrets Manager. Untuk informasi selengkapnya, lihat Mengkonfigurasi Peluncuran Cepat di Panduan AWS Marketplace Penjual.

Bagaimana AWS Migration Hub menggunakan AWS Secrets Manager

AWS Migration Hub menyediakan satu lokasi untuk melacak tugas migrasi di beberapa AWS alat dan solusi mitra.

AWS Migration Hub Orchestrator menyederhanakan dan mengotomatiskan migrasi server dan aplikasi perusahaan ke. AWS Migration Hub Orchestrator menggunakan rahasia untuk informasi koneksi ke server sumber Anda. Untuk informasi selengkapnya, di Panduan Pengguna AWS Migration Hub Orchestrator, lihat:

- Migrasikan aplikasi SAP ke NetWeaver AWS
- Rehost aplikasi di Amazon EC2

Rekomendasi Strategi Migration Hub menawarkan rekomendasi strategi migrasi dan modernisasi untuk jalur transformasi yang layak untuk aplikasi Anda. Rekomendasi Strategi dapat menganalisis database SQL Server, menggunakan rahasia untuk informasi koneksi. Untuk informasi selengkapnya, lihat analisis basis data Rekomendasi Strategi.

Bagaimana AWS Penggunaan Layanan Komputasi Paralel AWS Secrets Manager

AWS Parallel Computing Service (AWS PCS) adalah layanan terkelola yang memudahkan untuk menjalankan dan menskalakan komputasi kinerja tinggi (HPC) dan beban kerja machine learning yang didistribusikan. AWS

Untuk terhubung ke penjadwal pekerjaan cluster, AWS PCS membuat rahasia terkelola dengan awalan `pcs` untuk menyimpan kunci penjadwal. Biaya penyimpanan rahasia sudah termasuk dengan

biaya untuk AWS PCS. AWS PCS secara otomatis menghapus rahasia ketika Anda menghapus cluster AWS PCS Anda. Untuk informasi selengkapnya, lihat Bekerja dengan rahasia cluster di AWS PCS di Panduan Pengguna AWS PCS.

> **Important**
>
> Jangan memodifikasi atau menghapus rahasia klaster AWS PCS.

Bagaimana AWS ParallelCluster menggunakan AWS Secrets Manager

AWS ParallelCluster adalah alat manajemen cluster open source yang dapat Anda gunakan untuk menyebarkan dan mengelola cluster komputasi kinerja tinggi (HPC) di. AWS Cloud Anda dapat membuat beberapa lingkungan pengguna yang menyertakan yang terintegrasi dengan Microsoft AD AWS Terkelola (Direktori Aktif). AWS ParallelCluster AWS ParallelCluster Menggunakan rahasia Secrets Manager untuk memvalidasi login ke Active Directory. Untuk informasi selengkapnya, lihat Mengintegrasikan Active Directory di Panduan AWS ParallelCluster Pengguna.

Bagaimana Amazon Q menggunakan Secrets Manager

Untuk mengautentikasi Amazon Q untuk mengakses sumber data, Anda memberikan kredensyal akses sumber data ke Amazon Q menggunakan rahasia Secrets Manager. Jika Anda menggunakan konsol, Anda dapat memilih untuk membuat rahasia baru atau menggunakan yang sudah ada. Untuk informasi selengkapnya, lihat Konsep - Otentikasi di Panduan Pengembang Amazon Q.

Bagaimana Amazon OpenSearch Tertelan menggunakan Secrets Manager

Amazon OpenSearch Ingestionadalah pengumpul data tanpa server yang dikelola sepenuhnya yang mengalirkan log, metrik, dan data penelusuran waktu nyata ke domain dan koleksi OpenSearch Layanan Amazon. OpenSearch Serverless Anda dapat menggunakan OpenSearch Ingestion pipeline dengan Secrets Manager untuk mengelola kredensil Anda dengan aman. Untuk informasi lebih lanjut, lihat:

- Menggunakan OpenSearch Ingestion pipa dengan Atlassian Services
- Menggunakan OpenSearch Ingestion pipeline dengan Amazon DocumentDB
- Menggunakan OpenSearch Ingestion pipa dengan Confluent Cloud Kafka
- Menggunakan OpenSearch Ingestion pipa dengan Kafka

- Migrasi data dari cluster yang dikelola sendiri menggunakan OpenSearch Amazon OpenSearch Ingestion

Bagaimana AWS OpsWorks for Chef Automate menggunakan AWS Secrets Manager

OpsWorks adalah layanan manajemen konfigurasi yang membantu Anda mengkonfigurasi dan mengoperasikan aplikasi di perusahaan cloud dengan menggunakan OpsWorks untuk Puppet Enterprise atau AWS OpsWorks for Chef Automate.

Saat Anda membuat server baru AWS OpsWorks CM, OpsWorks CM menyimpan informasi untuk server dalam rahasia yang dikelola Secrets Manager dengan awalan`opsworks-cm`. Biaya rahasia sudah termasuk dalam biaya untuk OpsWorks. Untuk informasi selengkapnya, lihat Integrasi dengan AWS Secrets Manager di Panduan OpsWorks Pengguna.

Bagaimana Amazon Quick menggunakan AWS Secrets Manager

Amazon Quick adalah layanan intelijen bisnis skala cloud (BI) yang dapat Anda gunakan untuk analitik, visualisasi data, dan pelaporan. Anda dapat menggunakan berbagai sumber data di Quick. Jika Anda menyimpan kredensi database dalam rahasia Secrets Manager, Quick dapat menggunakan rahasia tersebut untuk terhubung ke database. Untuk informasi selengkapnya, lihat Menggunakan AWS Secrets Manager rahasia sebagai pengganti kredensi database di Amazon Quick di Panduan Pengguna Cepat Amazon.

Bagaimana Amazon RDS menggunakan AWS Secrets Manager

Amazon Relational Database Service (Amazon RDS) adalah layanan web yang mempermudah pengaturan, pengoperasian, dan penskalaan basis data relasional di AWS Cloud Cloud.

Untuk mengelola kredensi pengguna master untuk Amazon Relational Database Service (Amazon RDS), termasuk Aurora, Amazon RDS dapat membuat rahasia terkelola untuk Anda. Anda dituntut untuk rahasia itu. Amazon RDS juga mengelola rotasi untuk kredensil ini. Untuk informasi selengkapnya, lihat Manajemen kata sandi dengan Amazon RDS dan AWS Secrets Manager di Panduan Pengguna Amazon RDS.

Untuk kredensi Amazon RDS lainnya, lihat. Buat rahasia

Saat Anda menggunakan editor kueri Amazon RDS untuk menyambung ke database, Anda dapat menyimpan kredensyal untuk database di Secrets Manager. Untuk informasi selengkapnya, lihat Menggunakan editor kueri di Panduan Pengguna Amazon RDS.

Bagaimana Amazon Redshift menggunakan AWS Secrets Manager

Amazon Redshift adalah layanan gudang data dengan skala petabyte yang terkelola penuh di cloud.

Untuk mengelola kredensi admin untuk Amazon Redshift, Amazon Redshift dapat membuat rahasia terkelola untuk Anda. Anda dituntut untuk rahasia itu. Amazon Redshift juga mengelola rotasi untuk kredensi ini. Untuk informasi selengkapnya, lihat Mengelola kata sandi admin Amazon Redshift menggunakan AWS Secrets Manager dalam Panduan Manajemen Amazon Redshift.

Untuk kredenal Amazon Redshift lainnya, lihat. Buat rahasia

Saat memanggil Amazon Redshift Data API, Anda dapat meneruskan kredensyal untuk klaster dengan menggunakan rahasia di Secrets Manager. Untuk informasi selengkapnya, lihat Menggunakan Amazon Redshift Data API.

Saat Anda menggunakan editor kueri Amazon Redshift untuk terhubung ke database, Amazon Redshift dapat menyimpan kredensil Anda dalam rahasia Secrets Manager dengan awalan. `redshiftqueryeditor` Anda dituntut untuk rahasia itu. Untuk informasi selengkapnya, lihat Menanyakan database menggunakan editor kueri di Panduan Manajemen Amazon Redshift.

Untuk editor kueri v2, lihatthe section called “Editor kueri Amazon Redshift v2”.

Editor kueri Amazon Redshift v2

Editor kueri Amazon Redshift v2 adalah aplikasi klien SQL berbasis web yang dapat Anda gunakan untuk membuat dan menjalankan kueri di gudang data Amazon Redshift Anda. Saat Anda menggunakan editor kueri Amazon Redshift v2 untuk menyambung ke database, Amazon Redshift dapat menyimpan kredensil Anda dalam rahasia yang dikelola Secrets Manager dengan awalan. `sqlworkbench` Biaya penyimpanan rahasia sudah termasuk dengan biaya untuk menggunakan Amazon Redshift. Untuk memperbarui rahasia, Anda harus menggunakan Amazon Redshift daripada Secrets Manager. Untuk informasi selengkapnya, lihat Bekerja dengan editor kueri v2 di Panduan Manajemen Amazon Redshift.

Untuk editor kueri sebelumnya, lihatthe section called “Amazon Redshift”.

Bagaimana Amazon SageMaker AI menggunakan AWS Secrets Manager

SageMaker AI adalah layanan pembelajaran mesin yang dikelola sepenuhnya. Dengan SageMaker AI, ilmuwan dan pengembang data dapat dengan cepat dan mudah membangun dan melatih model pembelajaran mesin, dan kemudian langsung menerapkannya ke lingkungan host yang siap produksi. Ini menyediakan instance notebook pembuat Jupyter terintegrasi untuk akses mudah ke sumber data Anda untuk eksplorasi dan analisis, sehingga Anda tidak perlu mengelola server.

Anda dapat mengaitkan repositori Git dengan instance notebook Jupyter Anda untuk menyimpan buku catatan Anda di lingkungan kontrol sumber yang tetap ada meskipun Anda menghentikan atau menghapus instance buku catatan Anda. Anda dapat mengelola kredensi repositori pribadi Anda menggunakan Secrets Manager. Untuk informasi selengkapnya, lihat Mengaitkan Repositori Git dengan Instans SageMaker Notebook Amazon di Panduan Pengembang Amazon SageMaker AI.

Untuk mengimpor data dari Databricks, Data Wrangler menyimpan URL JDBC Anda di Secrets Manager. Untuk informasi selengkapnya, lihat Mengimpor data dari Databricks (JDBC).

Untuk mengimpor data dari Snowflake, Data Wrangler menyimpan kredensialmu dalam rahasia Secrets Manager. Untuk informasi selengkapnya, lihat Mengimpor data dari Snowflake.

Bagaimana AWS Schema Conversion Tool menggunakan AWS Secrets Manager

Anda dapat menggunakan AWS Schema Conversion Tool (AWS SCT) untuk mengonversi skema database yang ada dari satu mesin database ke mesin database lainnya. Anda dapat mengonversi skema OLTP relasional, atau skema gudang data. Skema konversi Anda cocok untuk Amazon Relational Database Service (Amazon RDS) MySQL, MariaDB, Oracle, SQL Server, PostgreSQL DB, kluster Amazon Aurora DB, atau cluster Amazon Redshift. Skema yang dikonversi juga dapat digunakan dengan database pada instans Amazon Elastic Compute Cloud atau disimpan sebagai data pada bucket S3.

Ketika Anda mengkonversi skema database, AWS SCT dapat menggunakan kredenal database yang Anda simpan di. AWS Secrets Manager Untuk informasi selengkapnya, lihat Menggunakan AWS Secrets Manager di antarmuka AWS SCT pengguna di Panduan AWS Schema Conversion Tool Pengguna.

Bagaimana Amazon Timestream untuk InfluxDB menggunakan AWS Secrets Manager

Timestream for InfluxDB adalah mesin database seri waktu terkelola yang memudahkan Anda menjalankan database InfluxDB untuk aplikasi deret waktu nyata menggunakan API sumber AWS terbuka. Dengan Timestream for InfluxDB, Anda dapat mengatur, mengoperasikan, dan menskalakan beban kerja deret waktu yang dapat menjawab kueri dengan waktu respons kueri milidetik satu digit.

Saat Anda membuat Timestream untuk database InfluxDB, Timestream secara otomatis membuat rahasia untuk menyimpan kredensyal admin. Untuk informasi selengkapnya, lihat Bagaimana Timestream for InfluxDB menggunakan rahasia di Panduan Pengembang Timestream.

Bagaimana AWS Toolkit for JetBrains menggunakan AWS Secrets Manager

AWS Toolkit for JetBrains Ini adalah plugin open source untuk lingkungan pengembangan terintegrasi (IDE) dari JetBrains. Toolkit memudahkan pengembang untuk mengembangkan, men-debug, dan menyebarkan aplikasi tanpa server yang digunakan. AWS Saat menghubungkan ke klaster Amazon Redshift menggunakan toolkit, Anda dapat mengautentikasi menggunakan rahasia Secrets Manager. Untuk informasi selengkapnya, lihat Mengakses kluster Amazon Redshift di AWS Toolkit for JetBrains Panduan Pengguna.

Bagaimana AWS Transfer Family menggunakan AWS Secrets Manager rahasia

AWS Transfer Family adalah layanan transfer aman yang memungkinkan Anda mentransfer file masuk dan keluar dari layanan AWS penyimpanan.

Transfer Family sekarang mendukung penggunaan otentikasi Dasar untuk server yang menggunakan protokol Applicability Statement 2 (AS2). Anda dapat membuat rahasia Secrets Manager baru atau memilih rahasia yang ada untuk kredensialmu. Untuk informasi selengkapnya, lihat Autentikasi dasar untuk konektor AS2 di AWS Transfer Family Panduan Pengguna.

Untuk mengautentikasi pengguna Transfer Family, Anda dapat menggunakannya AWS Secrets Manager sebagai penyedia identitas. Untuk informasi selengkapnya, lihat Bekerja dengan penyedia identitas kustom di Panduan AWS Transfer Family Pengguna dan artikel blog Aktifkan otentikasi kata sandi untuk AWS Transfer Family digunakan AWS Secrets Manager.

Anda dapat menggunakan dekripsi Pretty Good Privacy (PGP) dengan file yang diproses Transfer Family dengan alur kerja. Untuk menggunakan dekripsi dalam langkah alur kerja, Anda menyediakan kunci PGP yang Anda kelola di Secrets Manager. Untuk informasi selengkapnya, lihat Menghasilkan dan mengelola kunci PGP di AWS Transfer Family Panduan Pengguna.

Bagaimana AWS Wickr menggunakan AWS Secrets Manager rahasia

AWS Wickr adalah layanan terenkripsi end-to-end yang membantu organisasi dan lembaga pemerintah untuk berkomunikasi dengan aman melalui pesan satu-ke-satu dan grup, panggilan suara dan video, berbagi file, berbagi layar, dan banyak lagi. Anda dapat mengotomatiskan alur kerja menggunakan bot retensi data Wickr. Jika bot akan memiliki akses ke Layanan AWS, maka Anda harus membuat rahasia Secrets Manager untuk menyimpan kredensi bot. Untuk informasi selengkapnya, lihat Memulai bot retensi data di Panduan AWS Wickr Administrasi.

Penggunaan AWS Secrets Manager rahasia eksternal yang dikelola untuk mengelola rahasia Pihak Ketiga

Rahasia eksternal terkelola adalah jenis rahasia baru AWS Secrets Manager yang memungkinkan Anda menyimpan dan memutar kredensyal secara otomatis dari mitra integrasi. Fitur ini menghilangkan kebutuhan untuk membuat dan memelihara AWS Lambda fungsi khusus untuk memutar rahasia mitra integrasi. Untuk daftar lengkap semua mitra onboard, lihat Mitra Integrasi.

Saat Anda membangun aplikasi AWS, beban kerja Anda sering kali perlu berinteraksi dengan aplikasi pihak ketiga melalui kredensyal aman seperti kunci API, token OAuth, atau pasangan kredensi. Sebelumnya, Anda harus mengembangkan pendekatan khusus untuk mengamankan dan mengelola kredensyal ini, termasuk membangun fungsi Lambda rotasi kompleks yang unik untuk setiap aplikasi dan membutuhkan pemeliharaan berkelanjutan.

Rahasia eksternal terkelola menyediakan pendekatan standar untuk menyimpan kredensyal pihak ketiga dalam format yang telah ditentukan sebelumnya yang ditentukan oleh masing-masing mitra. Fitur ini mencakup rotasi otomatis yang diaktifkan (secara default di konsol) selama pembuatan rahasia, transparansi lengkap dan kontrol pengguna untuk alur kerja manajemen rahasia, dan set fitur lengkap yang ditawarkan oleh Secrets Manager termasuk manajemen izin halus, observabilitas, tata kelola, kepatuhan, pemulihan bencana, dan kontrol pemantauan.

Untuk mempelajari cara mengkonfigurasi rotasi otomatis untuk rahasia eksternal yang dikelola, menggunakan konsol atau AWS CLI, lihatRotate Secrets Manager mengelola rahasia eksternal. Untuk contoh langkah demi langkah yang menciptakan rahasia eksternal yang dikelola dan mengaktifkan rotasi, lihatBuat rahasia klien Salesforce.

Fitur utama

Rahasia eksternal yang dikelola menawarkan beberapa kemampuan utama yang menyederhanakan manajemen kredensyal pihak ketiga:

- Lambda-free rotasi terkelola menghilangkan overhead untuk membuat dan mengelola fungsi rotasi kustom. Saat Anda membuat eksternal, rotasi diaktifkan secara otomatis tanpa fungsi Lambda yang diterapkan di akun Anda.
- Format rahasia yang telah ditentukan memastikan bahwa rahasia dapat dikaitkan dengan benar dengan mitra integrasi dan menyertakan metadata yang diperlukan untuk rotasi. Setiap mitra mendefinisikan format yang diperlukan.

- Ekosistem mitra terintegrasi memberikan dukungan untuk beberapa mitra melalui proses orientasi standar. Mitra berintegrasi langsung dengan Secrets Manager untuk menawarkan panduan terprogram untuk pembuatan rahasia dan kemampuan rotasi terkelola.
- Auditabilitas lengkap mempertahankan transparansi penuh melalui AWS CloudTrail pencatatan untuk semua aktivitas rotasi, pembaruan nilai rahasia, dan operasi manajemen.

Rahasia eksternal yang dikelola Mitra

Secrets Manager terintegrasi secara native dengan aplikasi pihak ketiga untuk memutar rahasia yang dipegang oleh mitra. Setiap mitra mendefinisikan metadata dan bidang nilai rahasia yang diperlukan untuk memutar rahasia.

Nilai rahasia berisi bidang yang diperlukan untuk menghubungkan dengan klien pihak ketiga Anda dan disimpan selama CreateSecretpanggilan. Metadata rotasi memegang bidang yang digunakan untuk memperbarui rahasia selama rotasi dan digunakan dalam panggilan. RotateSecret Bidang ini akan ditentukan oleh mitra integrasi untuk memungkinkan aliran rotasi terkelola.

Agar rotasi berfungsi dengan baik, Anda harus memberikan Secrets Manager izin khusus untuk mengelola siklus hidup rahasia. Untuk informasi selengkapnya lihat Keamanan dan Izin

Topik berikut mencakup deskripsi masing-masing bidang metadata yang diperlukan untuk memutar rahasia serta deskripsi masing-masing bidang yang diperlukan dalam rahasia Secrets Manager untuk diputar.

Topik

Mitra Integrasi	Jenis rahasia
BIGID	BigIDClientSecret
Cisco	CiscoSecurityPlatformApiKey
Awan Konfluen	ConfluentCloudApiKey
Databricks	Databrickspat
Databricks	DatabrickspSecret
Datadog	DatadogApiKey

Mitra Integrasi	Jenis rahasia
Datadog	DatadogApplicationKey
Datadog	DatadogAdminKey
GitLab	GitLabAccessToken
Jenkins	JenkinsApiToken
MongoDB Atlas	MongoDBAtlasServiceAccount
MongoDB Atlas	MongoDBAtlasDatabaseUser
Netskope	NetskopeApiToken
Dayung	PaddleApiKey
Salesforce	SalesforceClientSecret
Snowflake	SnowflakeKeyPairAuthentication
Snowflake	SnowflakePat
SonarQube	SonarQubeToken
Twilio	TwilioApiKey
Twilio	TwilioAuthToken
Twilio	TwilioMainApiKey

Token Penyegaran ID Besar

Bidang Nilai Rahasia

Berikut ini adalah bidang-bidang yang harus dimuat dalam rahasia Secrets Manager:

```
{
  "hostname": "Host Name",
  "refreshToken": "Refresh Token"
```

```
}
```

hostname

Ini adalah nama host tempat instance BiGid Anda di-host. Anda harus memasukkan nama domain yang sepenuhnya memenuhi syarat dari instans Anda.

RefreshToken

Token penyegaran pengguna JWT yang dihasilkan di Konsol BiGid melalui Administrasi → Manajemen Akses → Pilih Pengguna → Hasilkan Token → Simpan

Aliran Penggunaan

Anda dapat membuat rahasia Anda menggunakan CreateSecretpanggilan dengan nilai rahasia yang berisi bidang yang disebutkan di atas dan tipe rahasia sebagai BigIDClientSecret. Konfigurasi rotasi dapat diatur menggunakan RotateSecretpanggilan. Anda juga harus memberikan peran ARN dalam RotateSecretpanggilan yang memberikan layanan izin yang diperlukan untuk memutar rahasia. Misalnya kebijakan izin, lihat Keamanan dan Izin. Perhatikan bahwa bidang metadata rotasi dapat dibiarkan kosong untuk mitra ini.

Cisco Kunci API Platform Keamanan

Bidang Nilai Rahasia

Berikut ini adalah bidang-bidang yang harus dimuat dalam rahasia Secrets Manager:

```
{
  "refreshToken": "refresh token value",
  "organizationId": "organization ID",
  "apiKeyId": "API key ID",
  "apiBaseUrl": "https://api.security.cisco.com"
}
```

RefreshToken

Kunci Cisco API, yang merupakan token penyegaran OAuth dengan masa pakai 60 hari. Proses rotasi memperbarui bidang ini. Aplikasi Anda menukar token ini dengan token akses berumur pendek.

OrganizationID

Organisasi Cisco Security Cloud Control yang memiliki kunci API. Nilai ini `enterpriseId` ditunjukkan di URL halaman Kunci API. Jangan gunakan ID organisasi yang disematkan dalam token.

api KeyId

Pengidentifikasi kunci API yang berputar. Anda dapat menemukan nilai ini di halaman Kunci API yang sama dengan ID organisasi.

api BaseUrl

URL dasar Cisco API, misalnya`https://api.security.cisco.com`. Nilai harus berupa asal HTTPS tanpa jalur, port, string kueri, atau informasi pengguna.

Rotasi juga menulis `expiresDate` bidang ke dalam nilai rahasia. Bidang ini mencatat kedaluwarsa token penyegaran saat ini dalam format ISO 8601. Ini bukan bidang yang Anda berikan.

Secrets Manager tidak menyimpan token akses. Token akses hidup sekitar 18 jam, jadi salinan yang disimpan akan menjadi tidak valid sebagian besar waktu. Aplikasi Anda mencetak token akses dari token penyegaran saat dibutuhkan.

Bidang Metadata Rahasia

Tipe rahasia ini tidak memerlukan bidang metadata rotasi. Secrets Manager membaca nilai routing dari secret itu sendiri, dan tipe rahasia ini tidak menggunakan rahasia admin.

Aliran Penggunaan

Rotasi ini menggunakan arsitektur rahasia tunggal. Tidak diperlukan rahasia admin. Token penyegaran yang disimpan mengotentikasi panggilan rotasinya sendiri.

Untuk membuat rahasia Anda, gunakan panggilan CreateSecretAPI. Tetapkan nilai rahasia ke bidang yang dijelaskan di atas dan atur tipe rahasia ke CiscoSecurityPlatformApiKey. Untuk mengonfigurasi rotasi, gunakan panggilan RotateSecretAPI. Dalam RotateSecret panggilan tersebut, berikan peran ARN yang memberikan izin layanan untuk memutar rahasia. Untuk contoh kebijakan izin, lihat Keamanan dan Izin.

Selama rotasi, Secrets Manager memanggil titik akhir penyegaran Cisco token dengan token penyegaran saat ini. Ciscomengembalikan token akses, dan setelah hari ke-45 dari siklus 60 hari itu juga mengembalikan token penyegaran baru. Secrets Manager menyimpan token penyegaran

terbaru sebagai versi yang tertunda dan membuang token akses. Kemudian memverifikasi token yang tertunda terhadap Cisco API dan mempromosikannya ke current. Tidak ada langkah pencabutan, karena tidak Cisco menyediakan API untuk membatalkan token penyegaran.

Aplikasi Anda membaca token penyegaran dari rahasia dan menukarnya dengan token akses. Kemudian menyimpan token akses itu selama 18 jam hidupnya. Baca token penyegaran dari rahasia setiap kali daripada melakukan hardcoding. Tangani respons HTTP 401 dengan membaca rahasianya lagi, karena token penyegaran sebelumnya mungkin berhenti berfungsi setelah Cisco mengeluarkan penggantian.

Putar rahasia ini setidaknya setiap 15 hari. Secrets Manager menolak interval rotasi lebih dari 50 hari, yang mengikuti Cisco panduan untuk menyegarkan setidaknya sesering itu. Interval yang lebih lama juga dapat melewati jendela antara hari 45 dan hari 60 di mana Cisco mengeluarkan token penyegaran pengganti. Jika sisa masa pakai token lebih pendek dari interval rotasi Anda, rotasi gagal dengan kesalahan alih-alih membiarkan token kedaluwarsa.

Token penyegaran yang kedaluwarsa tidak dapat diperpanjang. Jika itu terjadi, keluarkan kunci API baru di konsol Cisco Security Cloud Control dan perbarui rahasianya. Pantau kegagalan rotasi sehingga pemadaman yang berkepanjangan tidak mencapai keadaan itu.

Kunci Confluent Cloud API

Bidang Nilai Rahasia

Berikut ini adalah bidang-bidang yang harus dimuat dalam rahasia Secrets Manager:

```
{
  "apiKey": "API Key ID",
  "apiSecret": "API Secret",
  "serviceAccountId": "Service Account ID",
  "resourceId": "Resource ID",
  "environmentId": "Environment ID"
}
```

apiKey

ID Kunci Confluent Cloud API yang digunakan untuk otentikasi.

APISecret

Confluent Cloud API Secret digunakan untuk otentikasi.

layanan AccountId

ID prinsipal Akun Layanan yang diwakili oleh kunci API ini, misalnya`sa-abc123`. Logika rotasi menggunakan ini untuk membuat kunci baru untuk prinsipal yang benar.

RESOURCEID

(Opsional) ID sumber daya untuk pelingkupan kunci API. Ini bisa berupa cluster Kafka (`lkc-xxxxx`), kSqlDB cluster (`lksqlc-xxxxx`), Schema Registry (), wilayah Flink (,,`lsrc-xxxxx`), atau. `aws.us-west-2` `azure.centralus` `gcp.us-central1` `Tableflow` Hilangkan bidang ini untuk kunci API manajemen sumber daya cloud.

lingkunganTid

(Opsional) Confluent Cloud Environment ID, misalnya. `env-abcde` Digunakan saat membuat kunci cakupan cluster.

Bidang Metadata Rahasia

Berikut ini adalah bidang metadata untuk Confluent Cloud API Key:

```
{
  "adminSecretArn": "arn:aws:secretsmanager:us-
east-1:111122223333:secret:ConfluentCloudApiKey"
}
```

admin SecretArn

(Opsional) Amazon Resource Name (ARN) untuk rahasia yang berisi kredensyal Confluent Cloud API Key administratif yang digunakan untuk memutar rahasia ini. Kunci API admin harus memiliki CloudClusterAdmin atau OrganizationAdmin berperan untuk membuat dan menghapus kunci API untuk Akun Layanan. Jika dihilangkan, kredensyal rahasia pengguna sendiri digunakan untuk rotasi sendiri.

Aliran Penggunaan

Rotasi mendukung dua mode. Dalam mode rotasi mandiri (default), rahasia penggunaa`apiKey`/`apiSecret`digunakan untuk mengautentikasi panggilan Confluent API untuk pembuatan dan penghapusan kunci. Kunci API rahasia pengguna harus memiliki izin yang cukup

untuk mengelola kunci Akun Layanannya sendiri. Dalam mode admin-secret, rahasia admin terpisah yang berisi`apiKey`/`apiSecret`dengan izin admin digunakan sebagai gantinya.

Anda dapat membuat rahasia Anda menggunakan CreateSecretpanggilan dengan nilai rahasia yang berisi bidang yang disebutkan di atas dan tipe rahasia sebagai ConfluentCloudApiKey. Konfigurasi rotasi dapat diatur menggunakan RotateSecretpanggilan. Jika Anda memilih rotasi sendiri, Anda dapat menghilangkan bidang opsional`adminSecretArn`. Anda harus memberikan peran ARN dalam RotateSecretpanggilan yang memberikan layanan izin yang diperlukan untuk memutar rahasia. Untuk contoh kebijakan izin, lihat Keamanan dan Izin.

Untuk pelanggan yang memilih untuk memutar rahasia mereka menggunakan kumpulan kredensyal admin yang terpisah, buat Rahasia Admin AWS Secrets Manager yang berisi admin dan. `apiKey` `apiSecret` Anda harus memberikan ARN Rahasia Admin ini dalam metadata rotasi dalam RotateSecretpanggilan untuk rahasia kunci API Anda.

Selama rotasi, driver membuat kunci API baru untuk Akun Layanan target melalui Confluent Cloud API, memverifikasi kunci baru, memperbarui rahasia dengan kredensyal baru, dan menghapus kunci API lama.

Databricks Token Akses Pribadi

Bidang Nilai Rahasia

Berikut ini adalah bidang-bidang yang harus dimuat dalam rahasia Secrets Manager:

```
{
  "token": "dapi followed by 32 hexadecimal characters",
  "tokenId": "64-character hexadecimal token ID",
  "workspaceUrl": "https://dbc-xxxxxxxx-xxxx.cloud.databricks.com",
  "comment": "Comment shown in the token list"
}
```

token

Token akses Databricks pribadi. Secrets Manager memutar bidang ini.

TokenID

ID token. Secrets Manager menggunakannya untuk membaca cakupan token saat ini dan untuk mencabut token yang digantikannya.

WorkspaceURL

URL Databricks ruang kerja Anda, misalnya`https://dbc-xxxxxxxx-xxxx.cloud.databricks.com`. Berikan asal HTTPS kosong. Secrets Manager menolak URL yang berisi informasi pengguna, port, jalur, string kueri, fragmen, atau garis miring.

komentar

(Opsional) Komentar ditampilkan di samping token dalam daftar Databricks token. Panjang maksimalnya adalah 255 karakter.

Bidang Metadata Rahasia

Berikut ini adalah bidang metadata untuk Databricks Personal Access Token:

```
{
  "adminSecretArn": "arn:aws:secretsmanager:us-
east-1:111122223333:secret:DatabricksRotatorToken",
  "tokenLifetimeDays": "90"
}
```

admin SecretArn

(Opsional) Nama Sumber Daya Amazon (ARN) dari rahasia lain tipe DatabrickSpat yang memutar rahasia ini. Hilangkan bidang ini untuk membiarkan token berputar sendiri.

token LifetimeDays

(Opsional) Masa pakai setiap token baru, dalam beberapa hari. Nilai harus lebih besar dari interval rotasi Anda. Default: 90.

Administrator ruang kerja dapat menetapkan masa pakai token maksimum yang lebih rendah dari nilai yang Anda pilih di sini. Jika maksimum ruang kerja lebih rendah, pembuatan token gagal dan rotasi tidak dapat diselesaikan. Periksa pengaturan ruang kerja sebelum Anda memilih nilai.

Aliran Penggunaan

Jenis rotasi ini mendukung dua mode. Dalam rotasi diri, token berputar sendiri. Dalam rotasi berbantuan, token kedua dari jenis yang sama memutarnya. Dalam kedua kasus, token pengganti mewarisi cakupan token yang digantikannya. Rotasi gagal alih-alih membuat token dengan hak istimewa yang berbeda ketika Secrets Manager tidak dapat membaca cakupan tersebut.

Anda dapat membuat rahasia Anda menggunakan CreateSecretpanggilan dengan nilai rahasia yang berisi bidang yang disebutkan di atas dan jenis rahasia sebagai DatabrickSpat. Konfigurasi rotasi dapat diatur menggunakan RotateSecretpanggilan. Jika Anda memilih rotasi sendiri, Anda dapat menghilangkan bidang opsionaladminSecretArn. Anda harus memberikan peran ARN dalam RotateSecretpanggilan yang memberikan layanan izin yang diperlukan untuk memutar rahasia. Untuk contoh kebijakan izin, lihat Keamanan dan Izin.

Selama rotasi, Secrets Manager memvalidasi token saat ini dan membaca cakupannya. Ini menciptakan token pengganti yang membawa cakupan yang sama dan menyimpan pengganti sebagai AWSPENDING. Ini menegaskan bahwa token baru mengotentikasi. Ini juga menegaskan bahwa cakupan, pemilik, dan token baru cocok dengan token yang digantikannya. workspaceUrl Kemudian mencabut token sebelumnya dan mempromosikan AWSPENDING ke AWSCRENT.

Kedua token mengautentikasi sejak Secrets Manager membuat penggantian hingga mencabut token sebelumnya. Aplikasi yang membaca rahasia selama jendela itu tidak terganggu. Pencabutan itu sendiri pada akhirnya konsisten, karena Databricks cache keputusan otorisasi. Token sebelumnya mungkin terus diterima untuk waktu yang singkat setelah Secrets Manager mencabutnya. Saat Anda memutar sebagai respons terhadap dugaan kompromi kredensyal, konfirmasikan Databricks bahwa token tidak lagi diterima.

Databricksmencabut token akses pribadi yang tidak digunakan selama 90 hari. Rotasi menggunakan token rotator hanya ketika berjalan, jadi atur interval rotasi lebih pendek dari 90 hari. Jika tidakDatabricks, cabut token rotator karena tidak aktif dan rotasi selanjutnya gagal. Interval rotasi 30 hari dengan masa pakai default 90 hari memenuhi kedua persyaratan.

Secrets Manager menolak workspaceUrl yang menyelesaikan ke loopback, link-local, atau alamat pribadi. Ruang kerja yang Anda jangkau hanya melalui front-end AWS PrivateLink, dengan akses publik dinonaktifkan, oleh karena itu tidak didukung untuk rotasi token akses pribadi.

Databricks Rahasia Klien Utama Layanan OAuth

Bidang Nilai Rahasia

Berikut ini adalah bidang-bidang yang harus dimuat dalam rahasia Secrets Manager:

```
{
  "accountId": "Databricks account UUID",
  "servicePrincipalId": "Numeric service principal ID",
  "clientId": "Service principal application UUID",
  "clientSecret": "dose followed by the secret value",
```

```
  "oauthSecretId": "Hexadecimal OAuth secret ID"
}
```

accountId

ID Databricks akun Anda, yang merupakan UUID.

layanan PrincipalId

ID internal numerik dari kepala layanan. Nilai ini berbeda dari ID aplikasi.

clientId

ID aplikasi dari kepala layanan. Secrets Manager mengirimkannya seperti `client_id` dalam permintaan OAuth.

ClientSecret

Rahasia klien OAuth saat ini. Secrets Manager memutar bidang ini.

oauth SecretId

DatabricksID rahasia OAuth saat ini. Secrets Manager menggunakannya untuk membaca cakupan rahasia saat ini dan untuk menghapus rahasia yang digantikannya.

Bidang Metadata Rahasia

Berikut ini adalah bidang metadata untuk Databricks Service Principal OAuth Client Secret:

```
{
  "adminSecretArn": "arn:aws:secretsmanager:us-
east-1:111122223333:secret:DatabricksAdminServicePrincipal",
  "secretLifetimeDays": "730"
}
```

admin SecretArn

(Opsional) Nama Sumber Daya Amazon (ARN) dari rahasia lain tipe DatabricksspSecret yang memutar rahasia ini. Abaikan bidang ini untuk membiarkan kepala layanan memutar rahasianya sendiri.

rahasia LifetimeDays

(Opsional) Masa hidup setiap rahasia baru, dalam beberapa hari. Nilai harus lebih besar dari interval rotasi Anda. Maksimal adalah 730. Default: 730.

Aliran Penggunaan

Jenis rotasi ini mendukung dua mode. Dalam rotasi sendiri, kepala layanan memutar rahasianya sendiri. Dalam rotasi berbantuan, kepala layanan kedua memutarnya. Prinsip layanan yang melakukan rotasi membutuhkan `account_admin` peran, karena titik akhir rahasia OAuth adalah tingkat akun.

Anda dapat membuat rahasia Anda menggunakan CreateSecretpanggilan dengan nilai rahasia yang berisi bidang yang disebutkan di atas dan jenis rahasia sebagai DatabricksspSecret. Konfigurasi rotasi dapat diatur menggunakan RotateSecretpanggilan. Jika Anda memilih rotasi sendiri, Anda dapat menghilangkan bidang opsional`adminSecretArn`. Anda harus memberikan peran ARN dalam RotateSecretpanggilan yang memberikan layanan izin yang diperlukan untuk memutar rahasia. Untuk contoh kebijakan izin, lihat Keamanan dan Izin.

Selama rotasi, Secrets Manager meminta token akses menggunakan alur kredensyal klien. Ini menciptakan rahasia pengganti yang membawa cakupan rahasia yang digantikannya dan menyimpan penggantinya sebagai AWSPENDING. Ini menegaskan bahwa rahasia baru mengotentikasi. Kemudian menghapus rahasia sebelumnya dan mempromosikan AWSPENDING ke AWSCURRENT.

Seorang kepala layanan mendukung paling banyak lima rahasia OAuth. Jika kepala layanan sudah memegang lima, Databricks menolak permintaan dan rotasi gagal. Hapus rahasia yang tidak digunakan sebelum Anda memutar lagi.

`account_admin`Peran yang dibutuhkan tipe ini berbeda dari akses ruang kerja yang dibutuhkan rotasi Token Akses Databricks Pribadi. Hibah tunggal tidak mencakup kedua aliran.

Kunci Admin Datadog

Bidang Nilai Rahasia

Berikut ini adalah bidang-bidang yang harus dimuat dalam rahasia Secrets Manager:

```
{
  "adminApiKey": "32-character hex API key",
  "adminApiKeyId": "API key UUID",
  "adminAppKey": "Application key starting with ddapp_",
  "adminAppKeyId": "Application key UUID",
  "serviceAccountId": "Service Account UUID",
```

```
  "site": "datadoghq.com"
}
```

admin ApiKey

Kunci API admin Datadog (string heksadesimal 32 karakter).

admin ApiKeyId

Pengenal unik (UUID) untuk kunci API admin.

admin AppKey

Kunci Aplikasi admin Datadog. Harus dimiliki oleh akun layanan dan memiliki cakupan:`api_keys_write`,,`api_keys_delete`, `org_app_keys_read`org_app_keys_write,`service_account_write`.

admin AppKeyId

Pengenal unik (UUID) untuk kunci Aplikasi admin.

layanan AccountId

ID Akun Layanan Datadog (UUID) yang memiliki kunci Aplikasi admin.

situs

Situs Datadog Anda (misalnya,, `datadoghq.com`datadoghq.eu,`us5.datadoghq.com`).

Bidang Metadata Rahasia

Berikut ini adalah bidang metadata untuk Datadog Admin Key:

```
{
  "adminSecretArn": "arn:aws:secretsmanager:us-
east-1:111122223333:secret:DatadogAdminKey"
}
```

admin SecretArn

(Opsional) Nama Sumber Daya Amazon (ARN) untuk rahasia admin terpisah yang digunakan untuk otentikasi. Jika tidak disediakan, rahasia ini berputar sendiri menggunakan kredensialnya sendiri (rotasi diri).

Aliran Penggunaan

Jenis rotasi ini memutar kunci API dan kunci Aplikasi bersama-sama sebagai pasangan. Ini mendukung rotasi diri (default) di mana rahasia menggunakan kredensialnya sendiri untuk membuat penggantian, atau rotasi yang dibantu admin menggunakan rahasia admin terpisah.

Anda dapat membuat rahasia Anda menggunakan CreateSecretpanggilan dengan nilai rahasia yang berisi bidang yang disebutkan di atas dan tipe rahasia sebagai DatadogAdminKey. Konfigurasi rotasi dapat diatur menggunakan RotateSecretpanggilan. Jika Anda memilih rotasi sendiri, Anda dapat menghilangkan bidang opsional`adminSecretArn`. Anda harus memberikan peran ARN dalam RotateSecretpanggilan yang memberikan layanan izin yang diperlukan untuk memutar rahasia. Untuk contoh kebijakan izin, lihat Keamanan dan Izin.

Selama rotasi, driver memvalidasi kunci API saat ini, membuat kunci API baru dan kunci Aplikasi baru (mewarisi cakupan dari kunci saat ini), memverifikasi kedua kunci baru, menghapus pasangan lama menggunakan kredensyal baru, dan mempromosikan versi rahasia baru ke AWSCRENT.

Kunci API Datadog

Bidang Nilai Rahasia

Berikut ini adalah bidang-bidang yang harus dimuat dalam rahasia Secrets Manager:

```
{
  "apiKey": "32-character hex API key",
  "apiKeyId": "API key UUID"
}
```

apiKey

Kunci API Datadog saat ini. String heksadesimal 32 karakter yang digunakan untuk mengirimkan metrik, log, dan jejak ke Datadog.

api KeyId

Pengenal unik (UUID) untuk kunci API. Ditemukan melalui Datadog API atau Pengaturan Organisasi.

Bidang Metadata Rahasia

Berikut ini adalah bidang metadata untuk Datadog API Key:

```
{
  "adminSecretArn": "arn:aws:secretsmanager:us-
east-1:111122223333:secret:DatadogAdminKey"
}
```

admin SecretArn

Nama Sumber Daya Amazon (ARN) untuk rahasia tipe DatadogAdminKey yang berisi kredensyal Datadog administratif (kunci API dan kunci Aplikasi) yang digunakan untuk memutar rahasia ini. Kunci Aplikasi harus memiliki cakupan:`api_keys_write`,`api_keys_delete`.

Aliran Penggunaan

Rotasi ini menggunakan arsitektur dua rahasia. Jenis rahasia admin DatadogAdminKey menyediakan kunci API dan kunci Aplikasi yang diperlukan untuk mengautentikasi panggilan API Manajemen Kunci Datadog.

Anda dapat membuat rahasia Anda menggunakan CreateSecretpanggilan dengan nilai rahasia yang berisi bidang yang disebutkan di atas dan tipe rahasia sebagai DatadogApiKey. Konfigurasi rotasi dapat diatur menggunakan RotateSecretpanggilan. Anda harus memberikan metadata rotasi. `adminSecretArn` Anda juga harus memberikan peran ARN dalam RotateSecretpanggilan yang memberikan layanan izin yang diperlukan untuk memutar rahasia. Untuk contoh kebijakan izin, lihat Keamanan dan Izin.

Jenis rahasia admin (`DatadogAdminKey`) berbeda dari tipe rahasia pengguna (`DatadogApiKey`). Karena perbedaan ini, kebijakan peran rotasi default yang dicakup oleh tidak `secretsmanager:resource/Type` akan memberikan akses ke rahasia admin. Anda harus secara eksplisit memberikan akses peran rotasi ke rahasia admin. Anda dapat melakukan ini dengan menambahkan pernyataan yang dicakup ke jenisnya. `DatadogAdminKey` Atau, tentukan rahasia admin ARN secara langsung di kebijakan peran.

Selama rotasi, driver membuat kunci API baru melalui Datadog Key Management API v2, memverifikasi kunci baru menggunakan titik akhir validasi, mempromosikan kunci baru ke AWSCRENT, dan menghapus kunci yang dipindahkan (dua rotasi lama) dari Datadog. Ini mempertahankan pola bolak-balik 2-kunci yang memastikan rotasi zero-downtime.

Kunci Aplikasi Datadog

Bidang Nilai Rahasia

Berikut ini adalah bidang-bidang yang harus dimuat dalam rahasia Secrets Manager:

```
{
  "appKey": "Application key starting with ddapp_",
  "appKeyId": "Application key UUID",
  "serviceAccountId": "Service Account UUID"
}
```

AppKey

Kunci Aplikasi Datadog yang dimiliki oleh akun layanan. Dimulai dengan ddapp_ diikuti oleh 34 karakter alfanumerik.

aplikasi KeyId

Pengenal unik (UUID) untuk kunci Aplikasi.

layanan AccountId

ID Akun Layanan Datadog (UUID) yang memiliki kunci Aplikasi ini. Hanya akun layanan yang dimiliki Kunci aplikasi yang dapat diputar.

Bidang Metadata Rahasia

Berikut ini adalah bidang metadata untuk Datadog Application Key:

```
{
  "adminSecretArn": "arn:aws:secretsmanager:us-
east-1:111122223333:secret:DatadogAdminKey"
}
```

admin SecretArn

Nama Sumber Daya Amazon (ARN) untuk rahasia tipe DatadogAdminKey yang berisi kredensyal Datadog administratif (kunci API dan kunci Aplikasi) yang digunakan untuk memutar rahasia ini. Rahasia admin harus milik akun layanan yang sama dengan kunci Aplikasi ini.

Aliran Penggunaan

Rotasi ini menggunakan arsitektur dua rahasia. Jenis rahasia admin DatadogAdminKey menyediakan kredensyal otentikasi. Rahasia admin `serviceAccountId` harus sesuai dengan rahasia pengguna `serviceAccountId` untuk mencegah eskalasi hak istimewa.

Anda dapat membuat rahasia Anda menggunakan CreateSecretpanggilan dengan nilai rahasia yang berisi bidang yang disebutkan di atas dan tipe rahasia sebagai DatadogApplicationKey. Konfigurasi rotasi dapat diatur menggunakan RotateSecretpanggilan. Anda harus memberikan metadata rotasi. `adminSecretArn` Anda juga harus memberikan peran ARN dalam RotateSecretpanggilan yang memberikan layanan izin yang diperlukan untuk memutar rahasia. Untuk contoh kebijakan izin, lihat Keamanan dan Izin.

Jenis rahasia admin (`DatadogAdminKey`) berbeda dari tipe rahasia pengguna (`DatadogApplicationKey`). Karena perbedaan ini, kebijakan peran rotasi default yang dicakup oleh tidak `secretsmanager:resource/Type` akan memberikan akses ke rahasia admin. Anda harus secara eksplisit memberikan akses peran rotasi ke rahasia admin. Anda dapat melakukan ini dengan menambahkan pernyataan yang dicakup ke jenisnya. `DatadogAdminKey` Atau, tentukan rahasia admin ARN secara langsung di kebijakan peran.

Selama rotasi, driver memvalidasi kepemilikan kunci saat ini, membuat kunci Aplikasi baru melalui Datadog Service Account API, memverifikasi kunci baru, mempromosikannya ke AWSCRENT, dan menghapus kunci lama.

GitLab Token Akses

Bidang Nilai Rahasia

Berikut ini adalah bidang-bidang yang harus dimuat dalam rahasia Secrets Manager:

```
{
  "token": "GitLab access token value",
  "tokenId": "numeric token ID",
  "gitlabUrl": "GitLab instance URL",
  "projectId": "project ID (optional)",
  "groupId": "group ID (optional)"
}
```

token

Nilai token GitLab akses (dimulai dengan`glpat-`). Ini adalah bidang yang diputar.

TokenID

ID token numerik. Memperbarui setiap rotasi dengan ID token baru.

GitlaBurl

URL GitLab contoh Anda (misalnya,`https://gitlab.com`). Harus menggunakan HTTPS.

PROJECTID

(Opsional) ID proyek numerik. Sediakan token akses proyek saja.

groupId

(Opsional) ID grup numerik. Sediakan token akses grup saja.

Bidang Metadata Rahasia

Berikut ini adalah bidang metadata untuk GitLab Access Token:

```
{
  "adminSecretArn": "arn:aws:secretsmanager:us-east-1:111122223333:secret:GitLabAdmin",
  "daysToExpiry": "30 (optional)"
}
```

admin SecretArn

(Opsional) Nama Sumber Daya Amazon (ARN) untuk rahasia tipe GitLabAccessToken yang berisi token GitLab akses cakupan api yang digunakan untuk memutar rahasia ini. Jika dihilangkan, token berputar sendiri (membutuhkan `api` atau `self_rotate` ruang lingkup). Untuk token proyek, token admin memerlukan peran Maintainer pada proyek. Untuk token grup, diperlukan peran Pemilik di grup.

hari ToExpiry

(Opsional) Jumlah hari hingga token baru kedaluwarsa (1—365). Memetakan ke `expires_at` bidang di API GitLab putar. Jika dihilangkan, token baru mewarisi kedaluwarsa default instance.

Aliran Penggunaan

Rotasi ini mendukung arsitektur single-secret (self-rotation) dan two-secret (admin-assisted). Cakupan token ditentukan oleh opsional `projectId` dan `groupId` bidang. Jika tidak ada bidang

yang ada, token adalah token akses pribadi. Jika `projectId` ada, token adalah token akses proyek. Jika `groupId` ada, token adalah token akses grup.

Buat rahasia Anda menggunakan CreateSecretpanggilan. Tetapkan nilai rahasia ke bidang yang dijelaskan di atas dan atur tipe rahasia ke GitLabAccessToken. Untuk mengkonfigurasi rotasi, gunakan RotateSecretpanggilan. Berikan peran ARN yang memberikan layanan izin yang diperlukan untuk memutar rahasia. Untuk contoh kebijakan izin, lihat Keamanan dan Izin.

Saat menggunakan rotasi berbantuan admin, rahasia admin juga bertipe. `GitLabAccessToken` Anda harus secara eksplisit memberikan akses peran rotasi ke rahasia admin. Anda dapat melakukan ini dengan menambahkan pernyataan yang dicakup ke ARN rahasia admin langsung di kebijakan peran.

Selama rotasi, driver memvalidasi bahwa token saat ini aktif. Kemudian memanggil titik akhir GitLab putar, yang secara atom menciptakan token baru dan mencabut yang lama. Secrets Manager menyimpan nilai token dan ID baru sebagai AWSPENDING, memverifikasinya melalui GitLab API, dan mempromosikannya ke AWSCRENT. Aplikasi yang menggunakan pustaka caching Secrets Manager secara otomatis mengambil token baru pada penyegaran berikutnya.

Jenkins Token API

Bidang Nilai Rahasia

Berikut ini adalah bidang-bidang yang harus dimuat dalam rahasia Secrets Manager:

```
{
  "apiToken": "API token value",
  "tokenUuid": "token UUID",
  "username": "user that owns the token",
  "jenkinsUrl": "https://jenkins.example.com"
}
```

APIToken

Nilai token Jenkins API (string heksadesimal karakter 32-34). Proses rotasi memperbarui bidang ini.

TokenUUID

Pengidentifikasi unik universal (UUID) dari token API. Rotasi menggunakan UUID ini untuk mencabut token lama, dan memperbarui bidang ini dengan UUID token baru setelah setiap rotasi.

nama pengguna

JenkinsPengguna yang memiliki token.

JenkinSurl

URL Jenkins contoh Anda (misalnya,`https://jenkins.example.com`). URL harus menggunakan HTTPS dan tidak boleh diakhiri dengan garis miring.

Bidang Metadata Rahasia

Berikut ini adalah bidang metadata untuk Jenkins API Token:

```
{
  "adminSecretArn": "arn:aws:secretsmanager:us-east-1:111122223333:secret:JenkinsAdmin"
}
```

admin SecretArn

(Opsional) Nama Sumber Daya Amazon (ARN) untuk jenis rahasia JenkinsApiToken yang berisi token admin. Token admin dapat menghasilkan dan mencabut token API untuk pengguna target. Jika dihilangkan, token berputar sendiri (rotasi sendiri).

Aliran Penggunaan

Rotasi ini mendukung arsitektur single-secret (self-rotation) dan two-secret (admin-assisted). Saat Anda memberikan`adminSecretArn`, token admin menghasilkan dan mencabut token pengguna target. Jika Anda menghilangkannya, token saat ini berputar sendiri.

Untuk membuat rahasia Anda, gunakan panggilan CreateSecretAPI. Tetapkan nilai rahasia ke bidang yang dijelaskan di atas dan atur tipe rahasia ke JenkinsApiToken. Untuk mengonfigurasi rotasi, gunakan panggilan RotateSecretAPI. Dalam RotateSecret panggilan tersebut, berikan peran ARN yang memberikan izin layanan untuk memutar rahasia. Untuk contoh kebijakan izin, lihat Keamanan dan Izin. Saat menggunakan rotasi berbantuan admin, berikan akses peran rotasi secara eksplisit ke ARN rahasia admin.

Selama rotasi, driver memvalidasi token saat ini dan mendapatkan remah Cross-Site Request Forgery (CSRF) dari. Jenkins Kemudian menghasilkan token API baru dan menyimpannya sebagai versi yang tertunda. Ini memverifikasi token baru terhadap Jenkins API, mempromosikannya ke

saat ini, lalu mencabut token lama dengan miliknya. `tokenUuid` Aplikasi yang menggunakan perpustakaan caching Secrets Manager mengambil token baru pada penyegaran berikutnya.

Jenkinsdi-host pelanggan, jadi Jenkins instans Anda harus dapat dijangkau melalui HTTPS dari layanan rotasi Secrets Manager. Panggilan rotasi berasal dari daftar awalan AWS-managed. `com.amazonaws.region.secretsmanager-managed-external-secrets` Izinkan akses masuk dari daftar awalan ini di grup keamanan instans atau firewall Anda, dan pastikan instance menyajikan sertifikat TLS yang dipercaya publik. Untuk informasi selengkapnya, lihat daftar awalan AWS-terkelola di Panduan Pengguna Amazon VPC. Contoh yang tidak dapat dijangkau melalui internet publik tidak didukung.

Pengguna Database MongoDB Atlas

Bidang Nilai Rahasia

Berikut ini adalah bidang-bidang yang harus dimuat dalam rahasia Secrets Manager:

```
{
  "username": "database username",
  "password": "database password",
  "clusterUrl": "cluster hostname",
  "databaseName": "authentication database",
  "groupId": "Atlas Project ID"
}
```

nama pengguna

Nama pengguna database MongoDB (). SCRAM-authenticated Pengguna ini harus dikonfigurasi di MongoDB Atlas untuk menerima otentikasi SCRAM.

password

Kata sandi saat ini untuk pengguna database MongoDB Atlas.

ClusterURL

Nama host cluster MongoDB Atlas, misalnya. `cluster0.abc123.mongodb.net` Jangan sertakan `mongodb+srv://` awalan. Ini digunakan untuk memverifikasi kata sandi baru selama rotasi.

databaseName

Database otentikasi tempat kredensyal pengguna disimpan. Biasanya `admin` untuk pengguna SCRAM atau `$external` untuk X.509/LDAP.

groupId

ID Proyek Atlas heksadesimal 24 karakter (juga dikenal sebagai ID Grup). Anda dapat menemukan ini di Pengaturan Proyek Atlas Anda.

Bidang Metadata Rahasia

Berikut ini adalah bidang metadata untuk MongoDB Atlas Database User:

```
{
  "adminSecretArn": "arn:aws:secretsmanager:us-
east-1:111122223333:secret:MongoDBAtlasServiceAccount",
  "apiVersion": "2025-03-12"
}
```

admin SecretArn

Nama Sumber Daya Amazon (ARN) untuk rahasia yang berisi kredensyal OAuth akun layanan Atlas (tipe: MongoDBAtlasServiceAccount) dengan izin Admin Akses Database Project. Rahasia admin ini digunakan untuk mengautentikasi ke Atlas Admin API untuk pembaruan kata sandi.

apiVersion

(Opsional) Tanggal versi Atlas Admin API dalam `yyyy-mm-dd` format. Nilai ini digunakan di `Accept` header sebagai`application/vnd.atlas.{apiVersion}+json`. Default hingga `2025-03-12` jika tidak ditentukan.

Aliran Penggunaan

Jenis rotasi ini menggunakan arsitektur dua rahasia. Rahasia admin yang berisi kredensyal OAuth akun layanan Atlas (`clientId`,`clientSecret`,`serviceAccountId`) diperlukan untuk mengautentikasi ke Atlas Admin API. Rahasia admin harus dari tipe MongoDBAtlasServiceAccount.

Anda dapat membuat rahasia Anda menggunakan CreateSecretpanggilan dengan nilai rahasia yang berisi bidang yang disebutkan di atas dan tipe rahasia sebagai MongoDBAtlasDatabaseUser. Konfigurasi rotasi dapat diatur menggunakan RotateSecretpanggilan. Anda harus memberikan

metadata rotasi. `adminSecretArn` Anda juga harus memberikan peran ARN dalam RotateSecretpanggilan yang memberikan layanan izin yang diperlukan untuk memutar rahasia. Untuk contoh kebijakan izin, lihat Keamanan dan Izin.

Karena rahasia admin memiliki tipe (MongoDBAtlasServiceAccount) yang berbeda dari user secret (MongoDBAtlasDatabaseUser), kebijakan peran rotasi default yang dicakup oleh tidak `secretsmanager:resource/Type` akan memberikan akses ke rahasia admin. Anda harus secara eksplisit memberikan akses peran rotasi ke rahasia admin dengan menambahkan pernyataan yang dicakup ke MongoDBAtlasServiceAccount jenis atau dengan menentukan ARN rahasia admin secara langsung dalam kebijakan peran.

Selama rotasi, driver menghasilkan kata sandi baru, memanggil Atlas Admin API untuk memperbarui kata sandi pengguna database, dan memverifikasi kata sandi baru dengan membuka koneksi MongoDB nyata ke cluster. Perhatikan bahwa ada penundaan propagasi 5-10 detik setelah pembaruan kata sandi sebelum kata sandi baru diterima oleh lapisan otentikasi cluster.

Kredensi Akun Layanan MongoDB Atlas

Bidang Nilai Rahasia

Berikut ini adalah bidang-bidang yang harus dimuat dalam rahasia Secrets Manager:

```
{
  "clientId": "service account OAuth client ID",
  "clientSecret": "service account OAuth client secret",
  "orgId": "Atlas Organization ID"
}
```

clientId

ID klien OAuth akun layanan MongoDB Atlas. Ini harus dimulai dengan `mdb_sa_id_` diikuti oleh string heksadesimal 24-karakter.

ClientSecret

Rahasia klien OAuth akun layanan MongoDB Atlas digunakan untuk otentikasi.

OrgID

ID Organisasi Atlas heksadesimal 24 karakter. Anda dapat menemukannya di Pengaturan Organisasi Atlas Anda.

Bidang Metadata Rahasia

Berikut ini adalah bidang metadata untuk MongoDB Atlas Service Account:

```
{
  "adminSecretArn": "arn:aws:secretsmanager:us-
east-1:111122223333:secret:MongoDBAtlasServiceAccount",
  "apiVersion": "2025-03-12"
}
```

admin SecretArn

(Opsional) Nama Sumber Daya Amazon (ARN) untuk rahasia yang berisi kredensyal OAuth akun layanan administratif yang digunakan untuk memutar rahasia akun layanan ini. Rahasia admin harus berisi `clientSecret` nilai `clientId` dan dalam struktur rahasia. Jika dihilangkan, akun layanan akan menggunakan kredensialnya sendiri untuk rotasi sendiri.

apiVersion

(Opsional) Tanggal versi Atlas Admin API dalam `yyyy-mm-dd` format. Nilai ini digunakan di `Accept` header sebagai`application/vnd.atlas.{apiVersion}+json`. Default hingga `2025-03-12` jika tidak ditentukan.

Aliran Penggunaan

Rotasi mendukung dua mode otentikasi. Dalam mode rotasi diri (default), akun layanan menggunakan kredensialnya sendiri untuk membuat dan menghapus rahasianya. Ini mengharuskan akun layanan memiliki izin untuk mengelola rahasianya sendiri. Dalam mode rotasi berbantuan admin, kredenal akun layanan admin terpisah yang disimpan dalam rahasia lain digunakan. Ini diperlukan ketika akun layanan tidak memiliki izin manajemen mandiri.

Anda dapat membuat rahasia Anda menggunakan CreateSecretpanggilan dengan nilai rahasia yang berisi bidang yang disebutkan di atas dan tipe rahasia sebagai MongoDBAtlasServiceAccount. Konfigurasi rotasi dapat diatur menggunakan RotateSecretpanggilan. Jika Anda memilih rotasi sendiri, Anda dapat menghilangkan bidang opsional`adminSecretArn`. Anda harus memberikan peran ARN dalam RotateSecretpanggilan yang memberikan layanan izin yang diperlukan untuk memutar rahasia. Untuk contoh kebijakan izin, lihat Keamanan dan Izin.

Untuk pelanggan yang memilih untuk memutar rahasia mereka menggunakan kumpulan kredensyal terpisah (disimpan dalam Rahasia Admin), buat Rahasia Admin AWS Secrets Manager yang berisi

akun layanan admin dan. `clientId` `clientSecret` Anda harus memberikan ARN Rahasia Admin ini dalam metadata rotasi dalam RotateSecretpanggilan untuk rahasia akun layanan Anda.

Selama rotasi, driver membuat rahasia baru untuk akun layanan melalui Atlas Admin API, memverifikasi rahasia baru dengan membuat token OAuth, memperbarui rahasia dengan kredensyal baru, dan menghapus rahasia lama.

Netskope Token API

Bidang Nilai Rahasia

Berikut ini adalah bidang-bidang yang harus dimuat dalam rahasia Secrets Manager:

```
{
  "apiToken": "API token value",
  "tenantUrl": "https://example.goskope.com",
  "serviceAccountName": "service account name"
}
```

APIToken

Token pembawa Netskope REST API untuk akun layanan. Proses rotasi memperbarui bidang ini.

TenanTurl

URL Netskope penyewa Anda, misalnya`https://example.goskope.com`. URL harus menggunakan HTTPS, harus diakhiri`goskope.com`, dan tidak boleh diakhiri dengan garis miring. Host regional seperti `https://example.eu.goskope.com` juga valid.

layanan AccountName

Nama akun Netskope layanan yang memiliki token. Secrets Manager mencatat nama ini untuk logging dan audit.

Rotasi juga menulis `createdDate` dan `expiresDate` masuk ke nilai rahasia. Bidang ini mencatat waktu masalah dan kedaluwarsa token saat ini dalam format ISO 8601. Mereka bukan bidang yang Anda berikan.

Peran akun layanan tidak disimpan dalam rahasia. Peran tersebut milik akun layanan dan bukan token, jadi rotasi tidak pernah mengubahnya.

Bidang Metadata Rahasia

Tipe rahasia ini tidak memerlukan bidang metadata rotasi. Ini tidak menggunakan rahasia admin, dan Secrets Manager menetapkan masa pakai token menjadi 365 hari.

Aliran Penggunaan

Rotasi ini menggunakan arsitektur rahasia tunggal. Tidak diperlukan rahasia admin. Akun layanan menggunakan token saat ini untuk menghasilkan penggantinya sendiri.

Untuk membuat rahasia Anda, gunakan panggilan CreateSecretAPI. Tetapkan nilai rahasia ke bidang yang dijelaskan di atas dan atur tipe rahasia ke NetskopeApiToken. Untuk mengonfigurasi rotasi, gunakan panggilan RotateSecretAPI. Dalam RotateSecret panggilan tersebut, berikan peran ARN yang memberikan izin layanan untuk memutar rahasia. Untuk contoh kebijakan izin, lihat Keamanan dan Izin.

Selama rotasi, Secrets Manager memanggil titik akhir akun Netskope layanan dengan token saat ini untuk menghasilkan penggantian. Ini menyimpan token baru sebagai versi yang tertunda, memverifikasinya terhadap Netskope API, lalu mempromosikannya ke saat ini. Secrets Manager menolak interval rotasi lebih dari 350 hari, yang menjaga setiap interval dalam masa pakai token 365 hari.

Netskopemembatalkan token lama segera setelah menghasilkan yang baru, sehingga kedua token tidak pernah tumpang tindih. Selama kira-kira 30 detik selama rotasi, versi rahasia saat ini menyimpan token yang tidak lagi berfungsi. Aplikasi Anda harus menangani respons HTTP 401 dengan membaca rahasia lagi dan mencoba lagi. Aplikasi yang menyimpan token tanpa pola ini gagal selama setiap rotasi.

Jika peran akun layanan menggunakan daftar izin IP, tambahkan daftar awalan AWS-managed `com.amazonaws.region.secretsmanager-managed-external-secrets` ke allowlist tersebut. Panggilan rotasi berasal dari daftar awalan ini, dan Netskope mengembalikan HTTP 403 jika allowlist mengecualikannya. Untuk informasi selengkapnya, lihat daftar awalan AWS-terkelola di Panduan Pengguna Amazon VPC.

Kunci API Dayung

Bidang Nilai Rahasia

Berikut ini adalah bidang-bidang yang harus dimuat dalam rahasia Secrets Manager:

```
{
  "api_key": "Paddle API key value"
}
```

api_key

Nilai kunci Paddle API (dimulai dengan`pdl_live_`,`pdl_test_`, atau `pdl_sdbx_` diikuti oleh `apikey_` dan konten kunci). Ini adalah bidang yang diputar.

Bidang Metadata Rahasia

Berikut ini adalah bidang metadata untuk Paddle API Key:

```
{
  "gracePeriodSeconds": "300 (optional)"
}
```

rahmat PeriodSeconds

(Opsional) Jumlah detik kunci API lama tetap valid setelah rotasi (0—86400). Default: 300. Setel ke 0 untuk segera berakhir dari kunci lama. Hitung mundur masa tenggang dimulai ketika kunci baru pertama kali digunakan dengan sukses.

Aliran Penggunaan

Rotasi ini menggunakan arsitektur rahasia tunggal. Tidak diperlukan rahasia admin. Kunci API berputar sendiri melalui titik akhir rotasi Paddle.

Buat rahasia Anda menggunakan CreateSecretpanggilan. Tetapkan nilai rahasia ke bidang yang dijelaskan di atas dan atur tipe rahasia ke PaddleApiKey. Untuk mengkonfigurasi rotasi, gunakan RotateSecretpanggilan. Berikan peran ARN yang memberikan layanan izin yang diperlukan untuk memutar rahasia. Untuk contoh kebijakan izin, lihat Keamanan dan Izin.

Selama rotasi, driver mengekstrak ID kunci dan menyelesaikan host Paddle API dari key prefix. Kemudian memanggil titik akhir rotasi Paddle dengan masa tenggang yang dapat dikonfigurasi. Secrets Manager menyimpan kunci baru sebagai AWSPENDING, memverifikasinya dengan membuat panggilan API yang tidak bermutasi, dan mempromosikannya ke AWSCRENT. Kunci lama tetap berlaku selama masa tenggang (default 300 detik) sebelum kedaluwarsa otomatis.

Kunci API harus dibuat dengan opsi “dapat diputar” yang diaktifkan di Dasbor Dayung. Secrets Manager menolak kunci yang tidak dapat diputar dengan kesalahan.

Rahasia Klien Salesforce

Bidang Nilai Rahasia

Berikut ini adalah bidang-bidang yang harus dimuat dalam rahasia Secrets Manager:

```
{
  "consumerKey": "client ID",
  "consumerSecret": "client secret",
  "baseUri": "https://domain.my.salesforce.com",
  "appId": "app ID",
  "consumerId": "consumer ID"
}
```

CONSUMERKEY

Kunci konsumen, juga dikenal sebagai ID klien, adalah pengidentifikasi kredensyal untuk kredensyal OAuth 2.0. Anda dapat mengambil kunci konsumen langsung dari pengaturan OAuth Manajer Aplikasi Klien Eksternal Salesforce.

CONSUMERSECRET

Rahasia konsumen, juga dikenal sebagai rahasia klien, adalah kata sandi pribadi yang digunakan dengan kunci konsumen untuk mengautentikasi menggunakan alur kredensyal klien OAuth 2.0. Anda dapat mengambil rahasia konsumen langsung dari pengaturan OAuth Manajer Aplikasi Klien Eksternal Salesforce.

baseUri

URI dasar adalah URL dasar Org Salesforce Anda yang digunakan untuk berinteraksi dengan API Salesforce. Ini mengambil bentuk contoh berikut:`https://domainName.my.salesforce.com`.

APPID

App ID adalah pengenal untuk Salesforce External Client Application (ECA) Anda. Anda dapat mengambil ini dengan memanggil titik akhir Penggunaan OAuth Salesforce. Itu harus dimulai dengan `0x` dan hanya berisi karakter alfanumerik. Bidang ini mengacu pada external_client_app_identifier dalam panduan rotasi Salesforce.

ConsumerID

ID konsumen adalah pengenal untuk konsumen Salesforce External Client Application (ECA) Anda. Anda dapat mengambil ini dengan memanggil Salesforce OAuth Credentials by App ID endpoint. Bidang ini mengacu pada consumer_id dalam panduan rotasi Salesforce.

Bidang Metadata Rahasia

Berikut ini adalah bidang metadata yang diperlukan untuk memutar rahasia yang dipegang oleh Salesforce.

```
{
  "apiVersion": "v65.0",
  "adminSecretArn": "arn:aws:secretsmanager:us-
east-1:111122223333:secret:SalesforceClientSecret"
}
```

apiVersion

Versi API Salesforce adalah versi API organisasi Salesforce Anda. Versi harus setidaknya v65.0. Itu harus dalam format di *vXX.X* mana *X* adalah karakter numerik.

admin SecretArn

(Opsional) Rahasia admin ARN adalah Nama Sumber Daya Amazon (ARN) untuk rahasia yang berisi kredensyal OAuth administratif yang digunakan untuk memutar rahasia klien Salesforce ini. Minimal rahasia admin harus berisi nilai ConsumerKey dan ConsumerSecret dalam struktur rahasia. Ini adalah bidang opsional dan jika dihilangkan, selama rotasi Secrets Manager akan menggunakan kredensyal OAuth dalam rahasia ini untuk mengautentikasi dengan Salesforce.

Aliran Penggunaan

Dengan AWS Secrets Manager, Anda dapat memutar rahasia Salesforce menggunakan kredensil dalam rahasia yang sama atau menggunakan kredensyal dalam rahasia Admin untuk rotasi. Anda dapat membuat rahasia Anda menggunakan CreateSecretpanggilan dengan nilai rahasia yang berisi bidang yang disebutkan di atas dan tipe rahasia sebagai SalesforceClientSecret. Konfigurasi rotasi dapat diatur menggunakan RotateSecretpanggilan. Panggilan ini memerlukan spesifikasi bidang metadata seperti pada contoh di atas - Jika Anda memilih rotasi menggunakan kredensyal dalam rahasia yang sama, Anda dapat melewati bidang admin. SecretArn Selain itu, pelanggan

harus memberikan peran ARN dalam RotateSecretpanggilan yang memberikan layanan izin yang diperlukan untuk memutar rahasia. Untuk contoh kebijakan izin, lihat Keamanan dan Izin.

Untuk memutar rahasia Anda menggunakan kumpulan kredensyal terpisah yang disimpan dalam Rahasia Admin, buat Rahasia Admin dengan AWS Secrets Manager mengikuti langkah yang sama dengan rahasia konsumen Anda. Anda harus memberikan ARN Rahasia Admin ini dalam metadata rotasi dalam RotateSecretpanggilan untuk rahasia konsumen Anda.

Logika rotasi mengikuti panduan yang diberikan oleh Salesforce.

Pasangan Kunci Kepingan Salju

Bidang Nilai Rahasia

Berikut ini adalah bidang-bidang yang harus dimuat dalam rahasia Secrets Manager:

```
{
  "account": "Your Account Identifier",
  "user": "Your user name",
  "privateKey": "Your private Key",
  "publicKey": "Your public Key",
  "passphrase": "Your Passphrase"
}
```

user

Nama pengguna Snowflake yang terkait dengan otentikasi pasangan kunci ini. Pengguna ini harus dikonfigurasi di Snowflake untuk menerima otentikasi pasangan kunci, dan kunci publik harus ditetapkan ke profil pengguna ini.

akun

Pengidentifikasi akun Snowflake Anda digunakan untuk membuat koneksi. Ini dapat diekstraksi dari URL Snowflake Anda (bagian sebelum.snowflakecomputing.com)

PRIVATEKEY

Kunci pribadi RSA dalam format PEM digunakan untuk otentikasi. BEGIN/END Spidol bersifat opsional.

PublicKey

Mitra kunci publik dalam format PEM sesuai dengan kunci pribadi. BEGIN/END Spidol bersifat opsional.

frasa sandi

(Opsional) Bidang ini mengacu pada frasa sandi yang digunakan untuk mendekripsi kunci pribadi terenkripsi.

Bidang Metadata Rahasia

Berikut ini adalah bidang metadata untuk Snowflake:

```
{
  "cryptographicAlgorithm": "Your Cryptographic algorithm",
  "encryptPrivateKey": "True/False"
}
```

Algoritma Kriptografi

(Opsional) Ini mengacu pada algoritma yang digunakan untuk pembuatan kunci. Anda memiliki 3 pilihan algoritma:`RS256 | RS384 | RS512`. Bidang ini opsional dan algoritma default yang dipilih adalah RS256.

mengenkripsi PrivateKey

(Opsional) Bidang ini dapat digunakan untuk memilih apakah Anda ingin mengenkripsi kunci pribadi Anda. Nilainya secara default salah. Frasa sandi untuk enkripsi dihasilkan secara acak.

Aliran Penggunaan

Anda dapat membuat rahasia Anda menggunakan CreateSecretpanggilan dengan nilai rahasia yang berisi bidang yang disebutkan di atas dan tipe rahasia sebagai SnowflakeKeyPairAuthentication. Konfigurasi rotasi dapat diatur menggunakan RotateSecretpanggilan. Anda dapat secara opsional memberikan bidang metadata rahasia berdasarkan kebutuhan Anda. Anda juga harus memberikan peran ARN dalam RotateSecretpanggilan yang memberikan layanan izin yang diperlukan untuk memutar rahasia. Misalnya kebijakan izin, lihat Keamanan dan Izin. Perhatikan bahwa bidang metadata rotasi dapat dibiarkan kosong untuk mitra ini.

Token Akses Terprogram Kepingan Salju

Bidang Nilai Rahasia

Berikut ini adalah bidang-bidang yang harus dimuat dalam rahasia Secrets Manager:

```
{
  "account": "Snowflake account identifier",
  "user": "Snowflake username",
  "privateKey": "PEM-encoded private key",
  "passphrase": "private key passphrase (optional)",
  "patTokenName": "PAT name",
  "patTokenValue": "PAT secret value"
}
```

akun

Pengidentifikasi akun Snowflake Anda (misalnya,). `myorg-myaccount` Ini adalah bagian sebelumnya `.snowflakecomputing.com` di URL Snowflake Anda.

user

Nama pengguna Snowflake yang memiliki PAT. Pengguna ini harus memiliki otentikasi pasangan kunci yang dikonfigurasi.

PRIVATEKEY

PEM-encoded kunci pribadi untuk otentikasi pasangan kunci. Kunci ini tidak diputar - ini digunakan untuk mengautentikasi perintah ROTATE PAT (PAT tidak dapat memutar sendiri).

frasa sandi

(Opsional) Frasa sandi untuk kunci pribadi terenkripsi. Biarkan kosong jika kunci pribadi tidak dienkripsi.

tepuk TokenName

Nama token akses terprogram untuk diputar. Harus cocok dengan nama token di Snowflake.

tepuk TokenValue

Nilai rahasia token akses terprogram. Ini adalah bidang yang diputar.

Bidang Metadata Rahasia

Berikut ini adalah bidang metadata untuk Snowflake Programmatic Access Token:

```
{
  "daysToExpiry": "15",
  "expireOldTokenAfterHours": "24"
```

```
}
```

hari ToExpiry

(Opsional) Nilai DAYS_TO_EXPIRY PAT ditetapkan pada waktu pembuatan (1—365). Default: 15. Harus cocok dengan pengaturan Snowflake. Digunakan untuk memvalidasi bahwa jadwal rotasi lebih pendek dari TTL token.

kedaluwarsa OldTokenAfterHours

(Opsional) Beberapa jam sebelum token sebelumnya kedaluwarsa setelah rotasi (0—720). Default: 24. Setel ke 0 untuk segera kedaluwarsa token lama.

Aliran Penggunaan

Rotasi ini menggunakan arsitektur rahasia tunggal. Rahasianya berisi kredensyal pasangan kunci (untuk mengautentikasi perintah rotasi) dan nilai PAT (kredensyal yang diputar).

Anda dapat membuat rahasia Anda menggunakan CreateSecretpanggilan dengan nilai rahasia yang berisi bidang yang disebutkan di atas dan tipe rahasia sebagai SnowflakePat. Konfigurasi rotasi dapat diatur menggunakan RotateSecretpanggilan. Bidang metadata rotasi dapat dibiarkan kosong untuk menggunakan nilai default. Anda harus memberikan peran ARN dalam RotateSecretpanggilan yang memberikan layanan izin yang diperlukan untuk memutar rahasia. Untuk contoh kebijakan izin, lihat Keamanan dan Izin.

Selama rotasi, driver terhubung ke Snowflake melalui otentikasi pasangan kunci dan menjalankan `ALTER USER ... ROTATE PAT` perintah, yang secara atom menghasilkan token baru dan kedaluwarsa yang lama dengan masa tenggang yang dikonfigurasi. Token baru kemudian diverifikasi dengan menghubungkannya sebagai kata sandi.

SonarQube Token

Bidang Nilai Rahasia

Berikut ini adalah bidang-bidang yang harus dimuat dalam rahasia Secrets Manager:

```
{
  "token": "token value",
  "tokenName": "token name",
  "login": "user that owns the token",
  "sonarqubeUrl": "https://sonarqube.example.com",
```

```
  "tokenType": "USER_TOKEN",
  "projectKey": "project key"
}
```

token

Nilai SonarQube token (dimulai dengan`squ_`,`sqa_`, atau`sqp_`). Proses rotasi memperbarui bidang ini.

TokenName

Nama token (unik per pengguna). Rotasi menggunakan nama ini untuk mencabut token lama.

login

SonarQubePengguna yang memiliki token.

SonarqubeURL

URL SonarQube contoh Anda (misalnya,`https://sonarqube.example.com`). URL harus menggunakan HTTPS dan tidak boleh diakhiri dengan garis miring.

TokenType

(Opsional) Jenis token: `USER_TOKEN` (default),`GLOBAL_ANALYSIS_TOKEN`, atau`PROJECT_ANALYSIS_TOKEN`.

ProjectKey

(Opsional) Kunci SonarQube proyek. Diperlukan hanya bila `tokenType` ada`PROJECT_ANALYSIS_TOKEN`.

Bidang Metadata Rahasia

Berikut ini adalah bidang metadata untuk SonarQube Token:

```
{
  "adminSecretArn": "arn:aws:secretsmanager:us-
east-1:111122223333:secret:SonarQubeAdmin"
}
```

admin SecretArn

(Opsional) Nama Sumber Daya Amazon (ARN) untuk jenis rahasia SonarQubeToken yang berisi token admin. Bidang ini diperlukan untuk analisis global dan token Analisis Proyek. Ini opsional

untuk Token Pengguna; hilangkan untuk mengaktifkan rotasi sendiri, di mana token saat ini menghasilkan dan mencabut penggantinya sendiri.

Aliran Penggunaan

Rotasi ini mendukung rahasia tunggal (rotasi mandiri) untuk Token Pengguna dan rotasi dua-rahasia (dibantu admin) untuk analisis global dan token Analisis Proyek. `tokenType`Nilai menentukan cakupan token.

Untuk membuat rahasia Anda, gunakan panggilan CreateSecretAPI. Tetapkan nilai rahasia ke bidang yang dijelaskan di atas dan atur tipe rahasia ke SonarQubeToken. Untuk mengonfigurasi rotasi, gunakan panggilan RotateSecretAPI. Dalam RotateSecret panggilan tersebut, berikan peran ARN yang memberikan izin layanan untuk memutar rahasia. Untuk contoh kebijakan izin, lihat Keamanan dan Izin. Saat menggunakan rotasi berbantuan admin, berikan akses peran rotasi secara eksplisit ke ARN rahasia admin.

Selama rotasi, driver menghasilkan token baru, menyimpannya sebagai versi yang tertunda, memverifikasinya terhadap SonarQube API, mempromosikannya ke saat ini, lalu mencabut token lama dengan miliknya. `tokenName` Aplikasi yang menggunakan perpustakaan caching Secrets Manager mengambil token baru pada penyegaran berikutnya.

SonarQubeServer dihosting oleh pelanggan, jadi SonarQube instans Anda harus dapat dijangkau melalui HTTPS dari layanan rotasi Secrets Manager. Panggilan rotasi berasal dari daftar awalan AWS-managed. `com.amazonaws.region.secretsmanager-managed-external-secrets` Izinkan akses masuk dari daftar awalan ini di grup keamanan instans atau firewall Anda, dan pastikan instance menyajikan sertifikat TLS yang dipercaya publik. Untuk informasi selengkapnya, lihat daftar awalan AWS-terkelola di Panduan Pengguna Amazon VPC. Contoh yang tidak dapat dijangkau melalui internet publik tidak didukung.

Twilio Kunci API

Bidang Nilai Rahasia

Berikut ini adalah bidang-bidang yang harus dimuat dalam rahasia Secrets Manager:

```
{
  "accountSid": "AC followed by 32 hexadecimal characters",
  "apiKeySid": "SK followed by 32 hexadecimal characters",
```

```
  "apiKeySecret": "API key secret",
  "validationUrl": "https://api.twilio.com/2010-04-01/Accounts/ACxxxx/Messages.json?
PageSize=1"
}
```

AkunSID

TwilioAkun atau subaccount SID yang memiliki kunci API.

api KeySid

SID dari kunci API saat ini. Kunci Standar dan Terbatas didukung.

api KeySecret

Rahasia dipasangkan dengan SID kunci API saat ini. Secrets Manager memutar bidang ini bersama-sama dengan`apiKeySid`.

ValidationURL

(Opsional) URL Twilio API hanya-baca yang dipanggil Secrets Manager untuk mengonfirmasi kunci baru berfungsi. URL harus menggunakan HTTPS dan `twilio.com` host.

Bidang ini diperlukan untuk kunci Terbatas. Kunci Terbatas hanya dapat memanggil titik akhir yang diizinkan oleh kebijakannya, jadi berikan titik akhir yang Anda tahu diizinkan oleh kebijakan tersebut. Untuk kunci Standar, Secrets Manager mendapatkan titik akhir default.

Bidang Metadata Rahasia

Berikut ini adalah bidang metadata untuk Twilio API Key:

```
{
  "adminSecretArn": "arn:aws:secretsmanager:us-
east-1:111122223333:secret:TwilioMainApiKey",
  "retainPreviousKey": "false",
  "friendlyNamePrefix": "aws-sm-api-key"
}
```

admin SecretArn

Nama Sumber Daya Amazon (ARN) dari rahasia yang memegang kredensi administrator. Gunakan rahasia tipe TwilioMainApiKey atau rahasia tipe TwilioAuthToken.

mempertahankan PreviousKey

(Opsional) Apakah kunci sebelumnya tetap aktif setelah promosi. Default: false, yang segera menghapus kunci sebelumnya. Setel bidang ini ke true agar kunci sebelumnya dapat digunakan untuk satu siklus rotasi, sehingga aplikasi dapat menyegarkan kredensyal yang di-cache.

ramah NamePrefix

(Opsional) Awalan untuk nama ramah dari setiap tombol pengganti. Default: `aws-sm-api-key`.

Aliran Penggunaan

Rotasi ini menggunakan arsitektur dua rahasia. Kunci Twilio API tidak dapat berputar sendiri, karena titik akhir manajemen Twilio kunci menolak otentikasi kunci API. Oleh karena itu, rahasia administrator menyediakan kredensi yang membuat dan menghapus kunci.

Anda dapat membuat rahasia Anda menggunakan CreateSecretpanggilan dengan nilai rahasia yang berisi bidang yang disebutkan di atas dan tipe rahasia sebagai TwilioApiKey. Konfigurasi rotasi dapat diatur menggunakan RotateSecretpanggilan. Anda harus memberikan metadata rotasi. `adminSecretArn` Anda juga harus memberikan peran ARN dalam RotateSecretpanggilan yang memberikan layanan izin yang diperlukan untuk memutar rahasia. Untuk contoh kebijakan izin, lihat Keamanan dan Izin.

Jenis rahasia administrator (`TwilioMainApiKey`atau`TwilioAuthToken`) berbeda dari tipe rahasia pengguna (`TwilioApiKey`). Karena perbedaan ini, kebijakan peran rotasi default yang dicakup oleh `secretsmanager:resource/Type` tidak memberikan akses ke rahasia administrator. Anda harus secara eksplisit memberikan akses peran rotasi ke rahasia administrator. Anda dapat melakukan ini dengan menambahkan pernyataan yang dicakup ke tipe rahasia administrator. Atau, tentukan rahasia administrator ARN secara langsung dalam kebijakan peran.

Selama rotasi, Secrets Manager membaca jenis dan kebijakan kunci saat ini. Ini menciptakan kunci pengganti yang membawa jenis dan kebijakan yang sama, dan menyimpan penggantian sebagai AWSPENDING. Ini menegaskan bahwa kunci baru mengautentikasi terhadap titik akhir validasi. Kemudian menghapus kunci sebelumnya dan mempromosikan AWSPENDING ke AWSCRENT. Jika Anda menyetel `retainPreviousKey` ke true, Secrets Manager menyimpan kunci sebelumnya dan menghapusnya selama rotasi berikutnya.

Setiap kunci pengganti mewarisi tipe dan kebijakan kunci yang digantikannya, sehingga kunci Terbatas tetap Dibatasi dengan izin yang sama. Rotasi gagal alih-alih membuat kunci dengan hak

istimewa yang berbeda ketika Secrets Manager tidak dapat membaca jenis atau kebijakan kunci saat ini.

Twilio Token Auth

Bidang Nilai Rahasia

Berikut ini adalah bidang-bidang yang harus dimuat dalam rahasia Secrets Manager:

```
{
  "accountSid": "AC followed by 32 hexadecimal characters",
  "authToken": "32-character hexadecimal Auth Token",
  "validationUrl": "https://api.twilio.com/2010-04-01/Accounts/ACxxxx/Messages.json?
PageSize=1"
}
```

AkunSID

TwilioAkun SID yang Auth Token Secrets Manager berputar.

AuthToken

Token Auth utama saat ini untuk akun. Secrets Manager memutar bidang ini.

ValidationURL

(Opsional) URL Twilio API hanya-baca yang dipanggil Secrets Manager untuk mengonfirmasi token yang dipromosikan berfungsi. URL harus menggunakan HTTPS dan `twilio.com` host. Secrets Manager mendapatkan titik akhir default ketika Anda menghilangkan bidang ini.

Bidang Metadata Rahasia

Berikut ini adalah bidang metadata untuk Token Twilio Auth:

```
{
  "propagationDelaySeconds": "60"
}
```

propagasi DelaySeconds

(Opsional) Jumlah detik yang menunggu Secrets Manager sebelum memvalidasi Token Auth yang dipromosikan. Maksimumnya adalah 300. Default: 60, yang mencakup propagasi Twilio Fungsi.

Penundaan memperluas jendela di mana AWSCRENT menamai token sebelumnya. Jika aplikasi Anda tidak menggunakan Twilio Fungsi, atur bidang ini ke 0 untuk mempersingkat jendela itu.

Aliran Penggunaan

Jenis rotasi ini berputar sendiri. Secrets Manager mengautentikasi dengan Token Auth utama akun saat ini, sehingga tidak diperlukan kredensi administrator terpisah dan metadata rotasi tidak memiliki bidang wajib.

Anda dapat membuat rahasia Anda menggunakan CreateSecretpanggilan dengan nilai rahasia yang berisi bidang yang disebutkan di atas dan tipe rahasia sebagai TwilioAuthToken. Konfigurasi rotasi dapat diatur menggunakan RotateSecretpanggilan. Anda harus memberikan peran ARN dalam RotateSecretpanggilan yang memberikan layanan izin yang diperlukan untuk memutar rahasia. Untuk contoh kebijakan izin, lihat Keamanan dan Izin.

Selama rotasi, Secrets Manager membuat Token Auth sekunder dan menyimpannya sebagai AWSPENDING. Ini mempromosikan token sekunder ke primer, yang segera membatalkan token primer sebelumnya. Itu menunggu`propagationDelaySeconds`, mengonfirmasi bahwa token yang dipromosikan mengautentikasi, dan mempromosikan AWSPENDING ke AWSCRENT.

Promosi adalah apa yang membuat token baru dapat digunakan, dan itu membatalkan token sebelumnya pada saat yang sama. Oleh karena itu, tipe ini tidak memiliki jendela tumpang tindih di mana kedua token mengautentikasi. Aplikasi yang menyimpan Token Auth harus mengambilnya saat membuka koneksi, atau mencoba lagi setelah kegagalan otentikasi dengan membaca ulang AWSCRENT. Untuk jendela yang tumpang tindih, gunakan Twilio API Key dengan`retainPreviousKey`.

Twilio Kunci API Utama

Bidang Nilai Rahasia

Berikut ini adalah bidang-bidang yang harus dimuat dalam rahasia Secrets Manager:

```
{
  "accountSid": "AC followed by 32 hexadecimal characters",
  "adminKeySid": "SK followed by 32 hexadecimal characters",
  "adminKeySecret": "Main API key secret"
}
```

AkunSID

TwilioAkun SID yang memiliki Kunci API Utama.

admin KeySid

SID Kunci API Utama yang digunakan Secrets Manager sebagai kredensi administrator.

admin KeySecret

Rahasia dipasangkan dengan Main API Key SID.

Bidang Metadata Rahasia

Tipe rahasia ini tidak memiliki bidang metadata, karena Secrets Manager tidak memutarnya.

Aliran Penggunaan

Tipe rahasia ini menyimpan Kunci API Twilio Utama yang digunakan Secrets Manager untuk membuat dan menghapus kredensyal saat memutar rahasia tipe. TwilioApiKey Secrets Manager tidak memutar rahasia jenis ini.

Twiliotidak menyediakan operasi API yang membuat Kunci API Utama, sehingga Anda mengelola siklus hidup kredensyal ini di konsol. Twilio

Anda dapat membuat rahasia Anda menggunakan CreateSecretpanggilan dengan nilai rahasia yang berisi bidang yang disebutkan di atas dan tipe rahasia sebagai TwilioMainApiKey. Jangan mengkonfigurasi rotasi untuk rahasia ini. Referensi dari rahasia tipe TwilioApiKey melalui bidang metadata `adminSecretArn` rotasi.

Untuk menghindari kredensi administrator yang dikelola secara manual, `adminSecretArn` arahkan ke rahasia tipe TwilioAuthToken sebagai gantinya. Jenis itu berputar sendiri dan tidak memerlukan kredensi administratornya sendiri.

Keamanan dan izin

Rahasia eksternal yang dikelola tidak mengharuskan Anda untuk berbagi hak istimewa tingkat admin dari akun aplikasi pihak ketiga Anda. AWS Sebagai gantinya, proses rotasi menggunakan kredensyal dan metadata yang Anda berikan untuk melakukan panggilan API resmi ke aplikasi pihak ketiga untuk pembaruan dan validasi kredensyal.

Rahasia eksternal yang dikelola mempertahankan standar keamanan yang sama dengan jenis rahasia Secrets Manager lainnya. Nilai rahasia dienkripsi saat istirahat menggunakan kunci KMS Anda dan dalam perjalanan menggunakan TLS. Akses ke rahasia dikendalikan melalui kebijakan IAM dan kebijakan berbasis sumber daya. Saat menggunakan Kunci yang Dikelola Pelanggan untuk mengenkripsi rahasia Anda, Anda perlu memperbarui kebijakan IAM tentang peran rotasi dan kebijakan kepercayaan CMK untuk memberikan izin yang diperlukan untuk memastikan rotasi berhasil.

Agar rotasi berfungsi dengan baik, Anda harus memberikan Secrets Manager izin khusus untuk mengelola siklus hidup rahasia. Izin ini dapat dicakup ke rahasia individu dan mengikuti prinsip hak istimewa paling sedikit. Peran rotasi yang Anda berikan divalidasi selama penyiapan dan digunakan secara eksklusif untuk operasi rotasi.

Anda dapat membatasi masuknya IP ke sumber daya eksternal Anda dengan hanya mengizinkan daftar awalan terkelola rahasia AWS eksternal yang dikelola Secrets Manager di wilayah tempat rahasia Anda ada. Daftar awalan diberi nama`com.amazonaws.`*`region`*`.secretsmanager-managed-external-secrets`, di *`region`* mana AWS Wilayah rahasia Anda. Menggunakan daftar awalan terkelola memastikan aturan masuk Anda secara otomatis tetap terkini saat rentang IP yang mendasarinya berubah.

Jika Anda lebih suka mereferensikan daftar awalan secara terprogram, Anda dapat menggunakan GetManagedPrefixListEntriesAPI untuk membuat daftar IP yang perlu diijinkan. Anda harus me-refresh aturan ingress Anda secara berkala dengan daftar awalan ini agar tetap terkini.

AWS Secrets Manager juga menawarkan solusi sentuhan tunggal untuk membuat kebijakan IAM dengan izin yang diperlukan untuk mengelola rahasia saat membuat rahasia melalui konsol Secrets Manager. Izin untuk peran ini dicakup untuk setiap mitra integrasi di setiap wilayah.

Contoh Kebijakan Izin:

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowRotationAccess",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "secretsmanager:UpdateSecretVersionStage"
      ],
      "Resource": "*",
      "Effect": "Allow",
      "Condition": {
        "StringEquals": {
          "secretsmanager:resource/Type": "SalesforceClientSecret"
        }
      }
    },
    {
      "Sid": "AllowPasswordGenerationAccess",
      "Action": [
        "secretsmanager:GetRandomPassword"
      ],
      "Resource": "*",
      "Effect": "Allow"
    }
  ]
}
```

Catatan: Daftar tipe rahasia yang tersedia untuk secretsmanager: resource/Type dapat ditemukan di Mitra Integrasi.

Contoh Kebijakan Kepercayaan:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "SecretsManagerPrincipalAccess",
      "Effect": "Allow",
      "Principal": {
        "Service": "secretsmanager.amazonaws.com"
      },
```

```
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "111122223333"
        },
        "ArnLike": {
          "aws:SourceArn": "arn:aws:secretsmanager:us-east-1:111122223333:secret:*"
        }
      }
    }
  ]
}
```

Memantau dan memecahkan masalah rahasia eksternal yang dikelola

Rahasia eksternal terkelola menyediakan kemampuan pemantauan komprehensif melalui AWS CloudTrail log dan CloudWatch metrik Amazon. Semua aktivitas rotasi dicatat dengan informasi terperinci tentang keberhasilan, kegagalan, dan kesalahan apa pun yang ditemui selama proses.

Masalah umum dalam alur kerja rotasi termasuk konfigurasi izin peran yang salah atau nilai rahasia. Kegagalan untuk mengatur bidang ini adalah format yang ditentukan oleh mitra integrasi dapat menyebabkan kegagalan rotasi, karena layanan tidak akan dapat mengakses rahasia atau terhubung dengan klien mitra integrasi untuk memperbarui rahasia. Masalah lainnya bisa berupa masalah konektivitas jaringan, kedaluwarsa kredensyal, atau ketersediaan layanan mitra. Layanan rotasi terkelola mencakup logika coba ulang dan penanganan kesalahan untuk memaksimalkan keandalan

Anda dapat memantau jadwal rotasi, tingkat keberhasilan, dan metrik kinerja melalui Amazon. CloudWatch Anda dapat mengonfigurasi alarm khusus melalui jembatan acara untuk mengingatkan Anda tentang kegagalan rotasi atau masalah lain yang memerlukan perhatian.

Migrasi rahasia yang ada

Anda memiliki opsi untuk memigrasikan rahasia mitra yang ada ke rahasia eksternal yang dikelola. Ini bisa dilakukan dengan UpdateSecretpanggilan. Anda harus memperbarui nilai rahasia dan metadata seperti yang disebutkan dalam panduan. Jika Anda sudah memiliki logika rotasi khusus yang disiapkan untuk rahasia ini, Anda harus terlebih dahulu membatalkan rotasi menggunakan CancelRotateSecretpanggilan.

Pertimbangan dan batasan

Rahasia eksternal yang dikelola tidak mendukung rahasia fana dengan rentang hidup kurang dari empat jam. Rahasia yang terkait dengan sertifikat infrastruktur kunci publik juga tidak didukung.

Rahasia eksternal yang dikelola hanya didukung untuk mitra yang telah bergabung dengan. AWS Secrets Manager Untuk daftar lengkapnya, lihat Mitra Integrasi. Tidak melihat pasangan Anda dalam daftar? Beritahu mereka untuk Onboard untuk AWS Secrets Manager

Jika Anda memperbarui atau memutar nilai rahasia langsung dari layanan klien mitra di luar mesin rotasi Secrets Manager, sinkronisasi antar sistem dapat rusak. Meskipun Secrets Manager menyediakan peringatan konsol dan pencegahan terprogram untuk pembaruan nilai rahasia manual, Anda masih dapat memodifikasi nilai secara langsung di aplikasi pihak ketiga Anda. Untuk membangun kembali sinkronisasi setelah pembaruan out-of-band, Anda harus memperbarui nilai rahasia untuk mencerminkan rahasia yang benar dan kemudian memanggil RotateSecretAPI untuk memastikan rotasi sukses berkelanjutan.

Buat AWS Secrets Manager rahasia di AWS CloudFormation

Anda dapat membuat rahasia dalam CloudFormation tumpukan dengan menggunakan `AWS::SecretsManager::Secret` sumber daya dalam CloudFormation template, seperti yang ditunjukkan padaBuat rahasia.

Untuk membuat rahasia admin untuk Amazon RDS atau Aurora, kami sarankan Anda `ManageMasterUserPassword` menggunakannya. `AWS::RDS::DBCluster` Kemudian Amazon RDS menciptakan rahasia dan mengelola rotasi untuk Anda. Untuk informasi selengkapnya, lihat Rotasi terkelola.

Untuk kredensi Amazon Redshift dan Amazon DocumentDB, pertama-tama buat rahasia dengan kata sandi yang dihasilkan oleh Secrets Manager, dan kemudian gunakan referensi dinamis untuk mengambil nama pengguna dan kata sandi dari rahasia untuk digunakan sebagai kredensi untuk database baru. Selanjutnya, gunakan `AWS::SecretsManager::SecretTargetAttachment` sumber daya untuk menambahkan detail tentang database ke rahasia yang dibutuhkan Secrets Manager untuk memutar rahasia. Akhirnya, untuk mengaktifkan rotasi otomatis, gunakan `AWS::SecretsManager::RotationSchedule` sumber daya dan sediakan fungsi rotasi dan jadwal. Lihat contoh berikut:

- Buat rahasia dengan kredensi Amazon Redshift
- Buat rahasia dengan kredensi Amazon DocumentDB

Untuk melampirkan kebijakan sumber daya ke rahasia Anda, gunakan `AWS::SecretsManager::ResourcePolicy` sumber daya.

Untuk informasi tentang membuat sumber daya CloudFormation, lihat Mempelajari dasar-dasar templat di Panduan CloudFormation Pengguna. Anda juga dapat menggunakan AWS Cloud Development Kit (AWS CDK). Untuk informasi selengkapnya, lihat AWS Secrets Manager Membangun Perpustakaan.

Buat AWS Secrets Manager rahasia dengan CloudFormation

Contoh ini menciptakan rahasia bernama**`CloudFormationCreatedSecret-`*`a1b2c3d4e5f6`***. Nilai rahasianya adalah JSON berikut, dengan kata sandi 32 karakter yang dihasilkan saat rahasia dibuat.

```
{
    "password": "EXAMPLE-PASSWORD",
    "username": "saanvi"
}
```

Contoh ini menggunakan CloudFormation sumber daya berikut:

- AWS::SecretsManager::Secret

Untuk informasi tentang membuat sumber daya CloudFormation, lihat Mempelajari dasar-dasar templat di Panduan CloudFormation Pengguna.

JSON

```
{
    "Resources": {
        "CloudFormationCreatedSecret": {
            "Type": "AWS::SecretsManager::Secret",
            "Properties": {
                "Description": "Simple secret created by CloudFormation.",
                "GenerateSecretString": {
                    "SecretStringTemplate": "{\"username\": \"saanvi\"}",
                    "GenerateStringKey": "password",
                    "PasswordLength": 32
                }
            }
        }
    }
}
```

YAML

```
Resources:
  CloudFormationCreatedSecret:
    Type: 'AWS::SecretsManager::Secret'
    Properties:
      Description: Simple secret created by CloudFormation.
      GenerateSecretString:
        SecretStringTemplate: '{"username": "saanvi"}'
        GenerateStringKey: password
```

```
        PasswordLength: 32
```

Buat AWS Secrets Manager rahasia dengan rotasi otomatis dan instans Amazon RDS MySQL DB dengan CloudFormation

Untuk membuat rahasia admin untuk Amazon RDS atau Aurora, kami sarankan Anda `ManageMasterUserPassword` menggunakan, seperti yang ditunjukkan pada contoh Create a Secrets Manager rahasia untuk kata sandi utama di. `AWS::RDS::DBCluster` Kemudian Amazon RDS menciptakan rahasia dan mengelola rotasi untuk Anda. Lihat informasi yang lebih lengkap di Rotasi terkelola.

Buat AWS Secrets Manager rahasia dan cluster Amazon Redshift dengan CloudFormation

Untuk membuat rahasia admin untuk Amazon Redshift, kami sarankan Anda menggunakan contoh di `AWS::Redshift::Cluster`dan. `AWS::RedshiftServerless::Namespace`

Buat AWS Secrets Manager rahasia dan instance Amazon DocumentDB dengan CloudFormation

Contoh ini membuat rahasia dan instance Amazon DocumentDB menggunakan kredensi dalam rahasia sebagai pengguna dan kata sandi. Rahasianya memiliki kebijakan berbasis sumber daya terlampir yang mendefinisikan siapa yang dapat mengakses rahasia tersebut. Template juga membuat fungsi rotasi Lambda dari Templat fungsi rotasi dan mengonfigurasi rahasia untuk memutar secara otomatis antara pukul 08:00 dan 10:00 UTC pada hari pertama setiap bulan. Sebagai praktik keamanan terbaik, instancenya ada di VPC Amazon.

Contoh ini menggunakan CloudFormation sumber daya berikut untuk Secrets Manager:

- `AWS::SecretsManager::Secret`
- `AWS::SecretsManager::SecretTargetAttachment`
- `AWS::SecretsManager::RotationSchedule`

Untuk informasi tentang membuat sumber daya CloudFormation, lihat Mempelajari dasar-dasar templat di Panduan CloudFormation Pengguna.

JSON

```
{
   "AWSTemplateFormatVersion":"2010-09-09",
   "Transform":"AWS::SecretsManager-2020-07-23",
   "Resources":{
      "TestVPC":{
         "Type":"AWS::EC2::VPC",
         "Properties":{
            "CidrBlock":"10.0.0.0/16",
            "EnableDnsHostnames":true,
            "EnableDnsSupport":true
         }
      },
      "TestSubnet01":{
         "Type":"AWS::EC2::Subnet",
         "Properties":{
            "CidrBlock":"10.0.96.0/19",
            "AvailabilityZone":{
               "Fn::Select":[
                  "0",
                  {
                     "Fn::GetAZs":{
                        "Ref":"AWS::Region"
                     }
                  }
               ]
            },
            "VpcId":{
               "Ref":"TestVPC"
            }
         }
      },
      "TestSubnet02":{
         "Type":"AWS::EC2::Subnet",
         "Properties":{
            "CidrBlock":"10.0.128.0/19",
            "AvailabilityZone":{
               "Fn::Select":[
                  "1",
                  {
                     "Fn::GetAZs":{
                        "Ref":"AWS::Region"
```

```
                    }
                 }
              ]
           },
           "VpcId":{
              "Ref":"TestVPC"
           }
        }
     },
     "SecretsManagerVPCEndpoint":{
        "Type":"AWS::EC2::VPCEndpoint",
        "Properties":{
           "SubnetIds":[
              {
                 "Ref":"TestSubnet01"
              },
              {
                 "Ref":"TestSubnet02"
              }
           ],
           "SecurityGroupIds":[
              {
                 "Fn::GetAtt":[
                    "TestVPC",
                    "DefaultSecurityGroup"
                 ]
              }
           ],
           "VpcEndpointType":"Interface",
           "ServiceName":{
              "Fn::Sub":"com.amazonaws.${AWS::Region}.secretsmanager"
           },
           "PrivateDnsEnabled":true,
           "VpcId":{
              "Ref":"TestVPC"
           }
        }
     },
     "MyDocDBClusterRotationSecret":{
        "Type":"AWS::SecretsManager::Secret",
        "Properties":{
           "GenerateSecretString":{
              "SecretStringTemplate":"{\"username\": \"someadmin\",\"ssl\": true}",
              "GenerateStringKey":"password",
```

```
               "PasswordLength":16,
               "ExcludeCharacters":"\"@/\\"
            },
            "Tags":[
               {
                  "Key":"AppName",
                  "Value":"MyApp"
               }
            ]
         }
      },
      "MyDocDBCluster":{
         "Type":"AWS::DocDB::DBCluster",
         "Properties":{
            "DBSubnetGroupName":{
               "Ref":"MyDBSubnetGroup"
            },
            "MasterUsername":{
               "Fn::Sub":"{{resolve:secretsmanager:
${MyDocDBClusterRotationSecret}::username}}"
            },
            "MasterUserPassword":{
               "Fn::Sub":"{{resolve:secretsmanager:
${MyDocDBClusterRotationSecret}::password}}"
            },
            "VpcSecurityGroupIds":[
               {
                  "Fn::GetAtt":[
                     "TestVPC",
                     "DefaultSecurityGroup"
                  ]
               }
            ]
         }
      },
      "DocDBInstance":{
         "Type":"AWS::DocDB::DBInstance",
         "Properties":{
            "DBClusterIdentifier":{
               "Ref":"MyDocDBCluster"
            },
            "DBInstanceClass":"db.r5.large"
         }
      },
```

```
    "MyDBSubnetGroup":{
       "Type":"AWS::DocDB::DBSubnetGroup",
       "Properties":{
          "DBSubnetGroupDescription":"",
          "SubnetIds":[
             {
                "Ref":"TestSubnet01"
             },
             {
                "Ref":"TestSubnet02"
             }
          ]
       }
    },
    "SecretDocDBClusterAttachment":{
       "Type":"AWS::SecretsManager::SecretTargetAttachment",
       "Properties":{
          "SecretId":{
             "Ref":"MyDocDBClusterRotationSecret"
          },
          "TargetId":{
             "Ref":"MyDocDBCluster"
          },
          "TargetType":"AWS::DocDB::DBCluster"
       }
    },
    "MySecretRotationSchedule":{
       "Type":"AWS::SecretsManager::RotationSchedule",
       "DependsOn":"SecretDocDBClusterAttachment",
       "Properties":{
          "SecretId":{
             "Ref":"MyDocDBClusterRotationSecret"
          },
          "HostedRotationLambda":{
             "RotationType":"MongoDBSingleUser",
             "RotationLambdaName":"MongoDBSingleUser",
             "VpcSecurityGroupIds":{
                "Fn::GetAtt":[
                   "TestVPC",
                   "DefaultSecurityGroup"
                ]
             },
             "VpcSubnetIds":{
                "Fn::Join":[
```

```
                    ",",
                    [
                       {
                          "Ref":"TestSubnet01"
                       },
                       {
                          "Ref":"TestSubnet02"
                       }
                    ]
                 ]
              }
           },
           "RotationRules":{
             "Duration": "2h",
             "ScheduleExpression": "cron(0 8 1 * ? *)"
           }
        }
     }
  }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::SecretsManager-2020-07-23
Resources:
  TestVPC:
    Type: AWS::EC2::VPC
    Properties:
      CidrBlock: 10.0.0.0/16
      EnableDnsHostnames: true
      EnableDnsSupport: true
  TestSubnet01:
    Type: AWS::EC2::Subnet
    Properties:
      CidrBlock: 10.0.96.0/19
      AvailabilityZone: !Select
        - '0'
        - !GetAZs
          Ref: AWS::Region
      VpcId: !Ref TestVPC
  TestSubnet02:
    Type: AWS::EC2::Subnet
```

```
    Properties:
      CidrBlock: 10.0.128.0/19
      AvailabilityZone: !Select
        - '1'
        - !GetAZs
          Ref: AWS::Region
      VpcId: !Ref TestVPC
  SecretsManagerVPCEndpoint:
    Type: AWS::EC2::VPCEndpoint
    Properties:
      SubnetIds:
        - !Ref TestSubnet01
        - !Ref TestSubnet02
      SecurityGroupIds:
        - !GetAtt TestVPC.DefaultSecurityGroup
      VpcEndpointType: Interface
      ServiceName: !Sub com.amazonaws.${AWS::Region}.secretsmanager
      PrivateDnsEnabled: true
      VpcId: !Ref TestVPC
  MyDocDBClusterRotationSecret:
    Type: AWS::SecretsManager::Secret
    Properties:
      GenerateSecretString:
        SecretStringTemplate: '{"username": "someadmin","ssl": true}'
        GenerateStringKey: password
        PasswordLength: 16
        ExcludeCharacters: '"@/\'
      Tags:
        - Key: AppName
          Value: MyApp
  MyDocDBCluster:
    Type: AWS::DocDB::DBCluster
    Properties:
      DBSubnetGroupName: !Ref MyDBSubnetGroup
      MasterUsername: !Sub '{{resolve:secretsmanager:
${MyDocDBClusterRotationSecret}::username}}'
      MasterUserPassword: !Sub '{{resolve:secretsmanager:
${MyDocDBClusterRotationSecret}::password}}'
      VpcSecurityGroupIds:
        - !GetAtt TestVPC.DefaultSecurityGroup
  DocDBInstance:
    Type: AWS::DocDB::DBInstance
    Properties:
      DBClusterIdentifier: !Ref MyDocDBCluster
```

```
      DBInstanceClass: db.r5.large
  MyDBSubnetGroup:
    Type: AWS::DocDB::DBSubnetGroup
    Properties:
      DBSubnetGroupDescription: ''
      SubnetIds:
        - !Ref TestSubnet01
        - !Ref TestSubnet02
  SecretDocDBClusterAttachment:
    Type: AWS::SecretsManager::SecretTargetAttachment
    Properties:
      SecretId: !Ref MyDocDBClusterRotationSecret
      TargetId: !Ref MyDocDBCluster
      TargetType: AWS::DocDB::DBCluster
  MySecretRotationSchedule:
    Type: AWS::SecretsManager::RotationSchedule
    DependsOn: SecretDocDBClusterAttachment
    Properties:
      SecretId: !Ref MyDocDBClusterRotationSecret
      HostedRotationLambda:
        RotationType: MongoDBSingleUser
        RotationLambdaName: MongoDBSingleUser
        VpcSecurityGroupIds: !GetAtt TestVPC.DefaultSecurityGroup
        VpcSubnetIds: !Join
          - ','
          - - !Ref TestSubnet01
            - !Ref TestSubnet02
      RotationRules:
        Duration: 2h
        ScheduleExpression: cron(0 8 1 * ? *)
```

Bagaimana Secrets Manager menggunakan AWS CloudFormation

Saat Anda menggunakan konsol untuk mengaktifkan rotasi, Secrets Manager menggunakan AWS CloudFormation untuk membuat sumber daya untuk rotasi. Jika Anda membuat fungsi rotasi baru selama proses itu, CloudFormation buat `AWS::Serverless::Function`berdasarkan yang sesuaiTemplat fungsi rotasi. Kemudian CloudFormation atur `RotationSchedule`, yang mengatur fungsi rotasi dan aturan rotasi untuk rahasia. Anda dapat melihat CloudFormation tumpukan dengan memilih View stack di banner setelah Anda mengaktifkan rotasi otomatis.

Untuk informasi tentang mengaktifkan rotasi otomatis, lihatMerotasi rahasia .

Buat AWS Secrets Manager rahasia di AWS Cloud Development Kit (AWS CDK)

Untuk membuat, mengelola, dan mengambil rahasia di aplikasi CDK, Anda dapat menggunakan AWS Secrets Manager Construct Library, yang berisi,, `ResourcePolicy`, `RotationScheduleSecretSecretRotation`, dan konstruksi. `SecretTargetAttachment`

Praktik yang baik untuk menggunakan rahasia dalam aplikasi CDK adalah pertama-tama membuat rahasia dengan menggunakan konsol atau CLI, dan kemudian mengimpor rahasia ke dalam aplikasi CDK Anda.

Sebagai contoh, lihat:

- Buat rahasia
- Impor rahasia
- Ambil rahasia
- Berikan izin untuk menggunakan rahasia
- Putar rahasia
- Putar rahasia database
- Replikasi rahasia ke Wilayah lain

Untuk informasi selengkapnya tentang CDK, lihat Panduan Pengembang AWS Cloud Development Kit (AWS CDK) v2.

Pemberitahuan acara rahasia

Secrets Manager menerbitkan acara ke Amazon EventBridge saat label pementasan berubah pada rahasia Anda. Anda dapat membuat EventBridge aturan yang cocok dengan peristiwa ini dan meruteknya ke target untuk membangun alur kerja berbasis peristiwa.

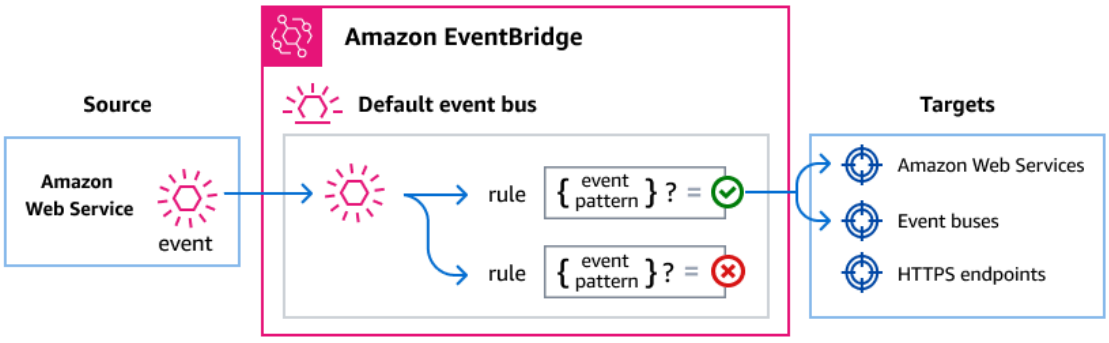
Cara EventBridge merutekan acara Secrets Manager

Berikut ini menjelaskan cara EventBridge kerja dengan peristiwa Secrets Manager.

Seperti banyak AWS layanan, Secrets Manager menghasilkan dan mengirim acara ke bus acara EventBridge default. Bus acara adalah router yang menerima acara dan merutekan mereka ke tujuan, atau target, yang Anda tentukan. Target dapat mencakup AWS layanan lain, aplikasi khusus, dan aplikasi mitra SaaS.

EventBridge rute acara sesuai dengan aturan yang Anda buat di bus acara. Untuk setiap aturan, Anda menentukan filter, atau pola acara, untuk memilih hanya peristiwa yang Anda inginkan. Setiap kali acara dikirim ke bus acara, EventBridge bandingkan dengan setiap aturan. Jika acara cocok dengan aturan, EventBridge rute acara ke target yang ditentukan.

Diagram berikut menunjukkan cara EventBridge merutekan peristiwa Secrets Manager dari bus peristiwa default ke target yang Anda tentukan.



Untuk informasi selengkapnya, lihat Membuat aturan di Panduan EventBridge Pengguna.

Acara Secrets Manager

Secrets Manager menerbitkan `Secret Label Updated` acara ke bus EventBridge acara default setiap kali label pementasan berubah, kecuali untuk `AWSPENDING` dan. `AWSPREVIOUS` Ini termasuk

label khusus. Acara diaktifkan secara default untuk semua rahasia. Untuk menggunakan peristiwa ini, Anda membuat EventBridge aturan yang cocok dengan mereka dan mengarahkannya ke target. Untuk daftar lengkap layanan yang dapat Anda gunakan sebagai target, lihat EventBridge target di Panduan EventBridge Pengguna.

Secrets Manager tidak mempublikasikan peristiwa untuk perubahan metadata seperti tag, deskripsi, konfigurasi rotasi, atau pembaruan kebijakan sumber daya.

Jenis detail acara	Deskripsi
Label Rahasia Diperbarui	Secrets Manager memindahkan label pementasan ke versi baru rahasia.

Struktur peristiwa

Semua peristiwa dari AWS layanan berisi dua jenis data:

- Seperangkat bidang umum yang berisi metadata tentang acara, seperti AWS layanan yang merupakan sumber acara, waktu acara dibuat, akun dan wilayah tempat acara berlangsung, dan lain-lain. Untuk definisi bidang umum ini, lihat Struktur acara di Referensi EventBridge Acara Amazon.
- `detail`Bidang yang berisi data khusus untuk acara layanan tertentu.

Membuat pola acara yang cocok dengan peristiwa Secrets Manager

Pola acara adalah filter tempat Anda menentukan data apa yang harus berisi peristiwa yang ingin Anda pilih.

Setiap pola acara adalah objek JSON yang berisi:

- `source`Atribut yang mengidentifikasi layanan yang mengirim acara. Untuk acara Secrets Manager, sumbernya adalah`aws.secretsmanager`.
- (Opsional): `detail-type` Atribut yang berisi array nama acara untuk dicocokkan.
- (Opsional): `detail` Atribut yang berisi data acara lain yang cocok.

Pola peristiwa berikut cocok dengan peristiwa saat `AWSCURRENT` label pindah ke versi baru, yang menunjukkan nilai rahasia aktif telah berubah:

```
{
  "source": ["aws.secretsmanager"],
  "detail-type": ["Secret Label Updated"],
  "detail": {
    "labelUpdated": ["AWSCURRENT"]
  }
}
```

Pola acara berikut cocok dengan perubahan label hanya untuk rahasia yang namanya dimulai dengan`MyTestSecret`:

```
{
  "source": ["aws.secretsmanager"],
  "detail-type": ["Secret Label Updated"],
  "detail": {
    "name": [{"prefix": "MyTestSecret"}]
  }
}
```

Pola acara berikut menggabungkan filter untuk dicocokkan hanya ketika `AWSCURRENT` perubahan pada rahasia dengan nama yang dimulai dengan`prod/`:

```
{
  "source": ["aws.secretsmanager"],
  "detail-type": ["Secret Label Updated"],
  "detail": {
    "name": [{"prefix": "prod/"}],
    "labelUpdated": ["AWSCURRENT"]
  }
}
```

Jika Anda menggunakan label pementasan khusus dalam alur kerja, Anda dapat mencocokkan peristiwa dengan label tersebut. Contoh berikut cocok ketika label bernama `MyLabel` dipindahkan:

```
{
  "source": ["aws.secretsmanager"],
  "detail-type": ["Secret Label Updated"],
  "detail": {
```

```
    "labelUpdated": ["MyLabel"]
  }
}
```

Untuk informasi selengkapnya tentang menulis pola peristiwa, lihat Pola acara di Panduan EventBridge Pengguna.

Membuat pola acara adalah langkah dalam proses pembuatan aturan yang lebih besar. Untuk informasi selengkapnya, lihat Membuat aturan di Panduan EventBridge Pengguna.

Label Rahasia Acara yang diperbarui

Secrets Manager mengirimkan `Secret Label Updated` acara ketika label pementasan pindah ke versi baru rahasia, untuk semua label kecuali `AWSPENDING` dan`AWSPREVIOUS`.

Contoh berikut menunjukkan struktur `Secret Label Updated` acara. Untuk deskripsi setiap bidang, lihat daftar berikut.

`detail-type`Bidang `source` dan disertakan di bawah ini karena berisi nilai khusus untuk acara Secrets Manager. Untuk definisi bidang metadata lain yang disertakan dalam semua peristiwa, lihat Struktur peristiwa di Referensi EventBridge Acara Amazon.

```
{
  "detail-type": "Secret Label Updated",
  "source": "aws.secretsmanager",
  "detail": {
    "name" : "string",
    "labelUpdated" : "string",
    "versionId" : "string"
  }
}
```

`detail-type`

Mengidentifikasi jenis acara.

Untuk acara ini, nilai ini adalah`Secret Label Updated`.

`source`

Mengidentifikasi layanan yang menghasilkan peristiwa. Untuk acara Secrets Manager, nilai ini adalah`aws.secretsmanager`.

`detail`

Objek JSON yang berisi informasi tentang peristiwa. Layanan yang menghasilkan acara menentukan konten bidang ini.

Untuk acara ini, data ini meliputi:

`name`

Nama pengguna yang mudah diingat.

`labelUpdated`

Label pementasan yang dilampirkan ke versi baru.

`versionId`

Pengidentifikasi unik dari versi baru rahasia.

Example Label Rahasia Acara yang diperbarui

Contoh berikut menunjukkan `Secret Label Updated` peristiwa untuk rahasia yang `AWSCURRENT` labelnya dipindahkan ke versi baru.

```
{
  "version": "0",
  "id": "6a7e8feb-b491-4cf7-a9f1-bf3703467718",
  "detail-type": "Secret Label Updated",
  "source": "aws.secretsmanager",
  "account": "012345678901",
  "time": "2024-02-06T16:43:48Z",
  "region": "us-west-2",
  "resources": [
    "arn:aws:secretsmanager:us-west-2:012345678901:secret:mySecret-a1b2c3"
  ],
  "detail": {
    "name": "mySecret",
    "labelUpdated": "AWSCURRENT",
    "versionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
  }
}
```

Memantau AWS Secrets Manager rahasia

AWS menyediakan alat pemantauan untuk menonton rahasia Secrets Manager, melaporkan ketika ada sesuatu yang salah, dan mengambil tindakan otomatis bila perlu. Anda dapat menggunakan log jika Anda perlu menyelidiki penggunaan atau perubahan yang tidak terduga, dan kemudian Anda dapat memutar kembali perubahan yang tidak diinginkan. Anda juga dapat mengatur pemeriksaan otomatis untuk penggunaan rahasia yang tidak pantas dan segala upaya untuk menghapus rahasia.

Topik

- Log AWS Secrets Manager acara dengan AWS CloudTrail
- Monitor AWS Secrets Manager dengan Amazon CloudWatch
- Pertandingan AWS Secrets Manager acara dengan Amazon EventBridge
- Pantau kapan AWS Secrets Manager rahasia yang dijadwalkan untuk dihapus diakses
- Memantau AWS Secrets Manager rahasia untuk kepatuhan dengan menggunakan AWS Config
- Biaya Monitor Secrets Manager
- Mendeteksi ancaman dengan Amazon GuardDuty

Log AWS Secrets Manager acara dengan AWS CloudTrail

AWS CloudTrail merekam semua panggilan API untuk Secrets Manager sebagai peristiwa, termasuk panggilan dari konsol Secrets Manager, serta beberapa peristiwa lain untuk rotasi dan penghapusan versi rahasia. Untuk daftar entri log di catatan Manajer Rahasia, lihatCloudTrail entri.

Anda dapat menggunakan CloudTrail konsol untuk melihat 90 hari terakhir dari peristiwa yang direkam. Untuk catatan peristiwa yang berkelanjutan di AWS akun Anda, termasuk peristiwa untuk Manajer Rahasia, buat jejak sehingga CloudTrail mengirimkan file log ke bucket Amazon S3. Lihat Membuat jejak untuk AWS akun Anda. Anda juga dapat mengonfigurasi CloudTrail untuk menerima file CloudTrail log dari beberapa Akun AWS dan Wilayah AWS.

Anda dapat mengonfigurasi AWS layanan lain untuk menganalisis lebih lanjut dan bertindak berdasarkan data yang dikumpulkan dalam CloudTrail log. Lihat integrasi AWS layanan dengan CloudTrail log. Anda juga bisa mendapatkan notifikasi saat CloudTrail menerbitkan file log baru ke bucket Amazon S3 Anda. Lihat Meng onfigurasi notifikasi Amazon SNS untuk CloudTrail.

Untuk mengambil peristiwa Manajer Rahasia dari CloudTrail log (konsol)

1. Buka CloudTrail konsol di https://console.aws.amazon.com/cloudtrail/.
2. Pastikan konsol menunjuk ke Wilayah tempat kejadian Anda terjadi. Konsol hanya menampilkan peristiwa yang terjadi di Wilayah yang dipilih. Pilih Wilayah dari daftar drop-down di sudut kanan atas konsol.
3. Di panel navigasi sebelah kiri, pilih Riwayat acara.
4. Pilih Filter kriteria and/or Rentang waktu untuk membantu Anda menemukan acara yang Anda cari. Contoh:

   a. Untuk melihat semua peristiwa Manajer Rahasia, untuk atribut Pencarian, pilih Sumber peristiwa. Kemudian, untuk Masukkan sumber peristiwa, pilih**`secretsmanager.amazonaws.com`**.

   b. Untuk melihat semua peristiwa untuk rahasia, untuk atribut Lookup, pilih Nama sumber daya. Kemudian, untuk Masukkan nama sumber daya, masukkan nama rahasia.
5. Untuk melihat detail tambahan, pilih panah perluas di sebelah acara. Untuk melihat semua informasi yang tersedia, pilih Lihat acara.

AWS CLI

Example Ambil peristiwa Manajer Rahasia dari CloudTrail log

`lookup-events`Contoh berikut mencari peristiwa Manajer Rahasia.

```
aws cloudtrail lookup-events \
    --region us-east-1 \
    --lookup-attributes
 AttributeKey=EventSource,AttributeValue=secretsmanager.amazonaws.com
```

AWS CloudTrail entri untuk Manajer Rahasia

AWS Secrets Manager menulis entri ke AWS CloudTrail log Anda untuk semua operasi Manajer Rahasia dan untuk acara lain yang terkait dengan rotasi dan penghapusan. Untuk informasi tentang mengambil tindakan pada acara ini, lihatAcara Match Secrets Manager dengan EventBridge.

Jenis entri log

- Entri log untuk operasi Secrets Manager

- Entri log untuk penghapusan
- Entri log untuk replikasi
- Entri log untuk rotasi

Entri log untuk operasi Secrets Manager

Peristiwa yang dihasilkan oleh panggilan ke operasi Manajer Rahasia memiliki`"detail-type": ["AWS API Call via CloudTrail"]`.

> **Note**
>
> Sebelum Februari 2024, beberapa operasi Manajer Rahasia melaporkan peristiwa yang berisi “ARN” alih-alih arn "" untuk ARN rahasia. Untuk informasi lebih lanjut, lihat AWS Re: Post.

Berikut ini adalah CloudTrail entri yang dihasilkan saat Anda atau layanan memanggil operasi Secrets Manager melalui API, SDK, atau CLI.

BatchGetSecretValue

Dihasilkan oleh BatchGetSecretValue operasi. Untuk informasi tentang mengambil rahasia, lihatDapatkan rahasia.

CancelRotateSecret

Dihasilkan oleh CancelRotateSecret operasi. Untuk informasi tentang rotasi, lihatMerotasi rahasia .

CreateSecret

Dihasilkan oleh CreateSecret operasi. Untuk informasi tentang membuat rahasia, lihatKelola rahasia.

DeleteResourcePolicy

Dihasilkan oleh DeleteResourcePolicy operasi. Untuk informasi tentang izin, lihatthe section called “Kontrol autentikasi dan akses”.

DeleteSecret

Dihasilkan oleh DeleteSecret operasi. Untuk informasi tentang menghapus rahasia, lihatthe section called “Hapus rahasia”.

DescribeSecret

Dihasilkan oleh DescribeSecret operasi.

GetRandomPassword

Dihasilkan oleh GetRandomPassword operasi.

GetResourcePolicy

Dihasilkan oleh GetResourcePolicy operasi. Untuk informasi tentang izin, lihatthe section called "Kontrol autentikasi dan akses".

GetSecretValue

Dihasilkan oleh BatchGetSecretValue operasi GetSecretValue dan. Untuk informasi tentang mengambil rahasia, lihatDapatkan rahasia.

ListSecrets

Dihasilkan oleh ListSecrets operasi. Untuk informasi tentang rahasia daftar, lihatthe section called "Temukan rahasia".

ListSecretVersionIds

Dihasilkan oleh ListSecretVersionIds operasi.

PutResourcePolicy

Dihasilkan oleh PutResourcePolicy operasi. Untuk informasi tentang izin, lihatthe section called "Kontrol autentikasi dan akses".

PutSecretValue

Dihasilkan oleh PutSecretValue operasi. Untuk informasi tentang memperbarui rahasia, lihatthe section called "Ubah rahasia".

RemoveRegionsFromReplication

Dihasilkan oleh RemoveRegionsFromReplication operasi. Untuk informasi tentang mereplikasi rahasia, lihatMulti-region replikasi.

ReplicateSecretToRegions

Dihasilkan oleh ReplicateSecretToRegions operasi. Untuk informasi tentang mereplikasi rahasia, lihatMulti-region replikasi.

RestoreSecret

Dihasilkan oleh RestoreSecret operasi. Untuk informasi tentang memulihkan rahasia yang dihapus, lihatthe section called “Kembalikan rahasia”.

RotateSecret

Dihasilkan oleh RotateSecret operasi. Untuk informasi tentang rotasi, lihatMerotasi rahasia .

StopReplicationToReplica

Dihasilkan oleh StopReplicationToReplica operasi. Untuk informasi tentang mereplikasi rahasia, lihatMulti-region replikasi.

TagResource

Dihasilkan oleh TagResource operasi. Untuk informasi tentang menandai rahasia, lihatthe section called “Kata kunci rahasia”.

UntagResource

Dihasilkan oleh UntagResource operasi. Untuk informasi tentang membuka tag rahasia, lihat. the section called “Kata kunci rahasia”

UpdateSecret

Dihasilkan oleh UpdateSecret operasi. Untuk informasi tentang memperbarui rahasia, lihatthe section called “Ubah rahasia”.

UpdateSecretVersionStage

Dihasilkan oleh UpdateSecretVersionStage operasi. Untuk informasi tentang tahapan versi, lihatthe section called “Versi rahasia”.

ValidateResourcePolicy

Dihasilkan oleh ValidateResourcePolicy operasi. Untuk informasi tentang izin, lihatthe section called “Kontrol autentikasi dan akses”.

Entri log untuk penghapusan

Selain peristiwa untuk operasi Manajer Rahasia, Manajer Rahasia menghasilkan peristiwa berikut terkait dengan penghapusan. Peristiwa ini memiliki`"detail-type": ["AWS Service Event via CloudTrail"]`.

CancelSecretVersionDelete

Dibuat oleh layanan Manajer Rahasia. Jika Anda memang `DeleteSecret` gil rahasia yang memiliki versi, dan kemudian memanggil nanti`RestoreSecret`, Manajer Rahasia mencatat peristiwa ini untuk setiap versi rahasia yang dipulihkan. Untuk informasi tentang memulihkan rahasia yang dihapus, lihatthe section called “Kembalikan rahasia”.

EndSecretVersionDelete

Dibuat oleh layanan Manajer Rahasia ketika versi rahasia dihapus. Untuk informasi selengkapnya, lihat the section called “Hapus rahasia”.

StartSecretVersionDelete

Dibuat oleh layanan Manajer Rahasia ketika Manajer Rahasia mulai dihapus untuk versi rahasia. Untuk informasi tentang menghapus rahasia, lihatthe section called “Hapus rahasia”.

SecretVersionDeletion

Dibuat oleh layanan Manajer Rahasia saat Manajer Rahasia menghapus versi rahasia yang tidak digunakan lagi. Untuk informasi selengkapnya, lihat Versi rahasia.

Entri log untuk replikasi

Selain peristiwa untuk operasi Manajer Rahasia, Manajer Rahasia menghasilkan peristiwa berikut yang terkait dengan replikasi. Peristiwa ini memiliki`"detail-type": ["AWS Service Event via CloudTrail"]`.

ReplicationFailed

Dibuat oleh layanan Secrets Manager saat replikasi gagal. Untuk informasi tentang mereplikasi rahasia, lihatMulti-region replikasi.

ReplicationStarted

Dibuat oleh layanan Manajer Rahasia ketika Manajer Rahasia mulai mereplikasi rahasia. Untuk informasi tentang mereplikasi rahasia, lihatMulti-region replikasi.

ReplicationSucceeded

Dibuat oleh layanan Manajer Rahasia ketika rahasia berhasil direplikasi. Untuk informasi tentang mereplikasi rahasia, lihatMulti-region replikasi.

Entri log untuk rotasi

Selain peristiwa untuk operasi Manajer Rahasia, Manajer Rahasia menghasilkan peristiwa berikut terkait dengan rotasi. Peristiwa ini memiliki`"detail-type": ["AWS Service Event via CloudTrail"]`.

RotationStarted

Dibuat oleh layanan Manajer Rahasia ketika Manajer Rahasia mulai memutar rahasia. Untuk informasi tentang rotasi, lihatMerotasi rahasia .

RotationAbandoned

Dibuat oleh layanan Manajer Rahasia ketika Manajer Rahasia meninggalkan upaya rotasi dan menghapus `AWSPENDING` label dari versi rahasia yang ada. Manajer Rahasia meninggalkan rotasi saat Anda membuat versi baru rahasia selama rotasi. Untuk informasi tentang rotasi, lihatMerotasi rahasia .

RotationFailed

Dibuat oleh layanan Manajer Rahasia saat rotasi gagal. Untuk informasi tentang rotasi, lihatthe section called “Memecahkan masalah rotasi”.

RotationSucceeded

Dibuat oleh layanan Manajer Rahasia ketika rahasia berhasil diputar. Untuk informasi tentang rotasi, lihatMerotasi rahasia .

TestRotationStarted

Dibuat oleh layanan Manajer Rahasia ketika Manajer Rahasia mulai menguji rotasi untuk rahasia yang tidak dijadwalkan untuk rotasi segera. Untuk informasi tentang rotasi, lihatMerotasi rahasia .

TestRotationSucceeded

Dibuat oleh layanan Manajer Rahasia ketika Manajer Rahasia berhasil menguji rotasi untuk rahasia yang tidak dijadwalkan untuk rotasi segera. Untuk informasi tentang rotasi, lihatMerotasi rahasia .

TestRotationFailed

Dibuat oleh layanan Manajer Rahasia ketika Manajer Rahasia menguji rotasi untuk rahasia yang tidak dijadwalkan untuk rotasi langsung dan rotasi gagal. Untuk informasi tentang rotasi, lihatthe section called “Memecahkan masalah rotasi”.

Monitor AWS Secrets Manager dengan Amazon CloudWatch

Menggunakan Amazon CloudWatch, Anda dapat memantau AWS layanan dan membuat alarm untuk memberi tahu Anda saat metrik berubah. CloudWatch menyimpan statistik ini selama 15 bulan, sehingga Anda dapat mengakses informasi historis dan mendapatkan perspektif yang lebih baik tentang kinerja aplikasi atau layanan web Anda. Untuk AWS Secrets Manager, Anda dapat memantau jumlah rahasia di akun Anda, termasuk rahasia yang ditandai untuk dihapus, dan panggilan API ke Secrets Manager, termasuk panggilan yang dilakukan melalui konsol. Untuk informasi tentang cara memantau metrik, lihat Menggunakan CloudWatch metrik di CloudWatch Panduan Pengguna.

Untuk menemukan metrik Secrets Manager

1. Di CloudWatch konsol, di bawah Metrik, pilih Semua metrik.
2. Dalam pencarian Metrik, kotak, masukkan`secret`.
3. Lakukan hal-hal berikut:
   - Untuk memantau jumlah rahasia di akun Anda, pilih AWS/SecretsManager, lalu pilih SecretCount. Metrik ini diterbitkan setiap jam.
   - Untuk memantau panggilan API ke Secrets Manager, termasuk panggilan yang dilakukan melalui konsol, pilih Usage > By AWS Resource, lalu pilih panggilan API yang akan dipantau. Untuk mengetahui daftar Secrets Manager API, lihat Operasi Secrets Manager.
4. Lakukan hal-hal berikut:
   - Untuk membuat grafik metrik, lihat Metrik grafik di CloudWatch Panduan Pengguna Amazon.
   - Untuk mendeteksi anomali, lihat Menggunakan deteksi CloudWatch anomali di Panduan Pengguna Amazon. CloudWatch
   - Untuk mendapatkan statistik untuk metrik, lihat Mendapatkan statistik untuk metrik di Panduan CloudWatch Pengguna Amazon.

CloudWatch alarm

Anda dapat membuat CloudWatch alarm yang mengirimkan pesan Amazon SNS saat nilai metrik berubah dan menyebabkan alarm berubah status. Anda dapat mengatur alarm pada metrik Secrets Manager`ResourceCount`, yang merupakan jumlah rahasia di akun Anda. Anda juga dapat mengatur alarm pada Alarm mengawasi metrik selama periode waktu yang Anda tentukan, dan melakukan

tindakan berdasarkan nilai metrik relatif terhadap ambang batas tertentu selama beberapa periode waktu. Alarm memanggil tindakan untuk perubahan status berkelanjutan saja. CloudWatch alarm tidak memanggil tindakan hanya karena mereka berada dalam keadaan tertentu; negara harus telah berubah dan dipertahankan untuk sejumlah periode tertentu.

Untuk informasi selengkapnya, lihat Menggunakan CloudWatch alarm Amazon dan Membuat CloudWatch alarm berdasarkan deteksi anomali di Panduan Pengguna. CloudWatch

Anda juga dapat mengatur alarm yang memperhatikan ambang batas tertentu dan mengirim notifikasi atau mengambil tindakan saat ambang batas tersebut terpenuhi. Untuk informasi selengkapnya, lihat Panduan CloudWatch Pengguna Amazon.

Pertandingan AWS Secrets Manager acara dengan Amazon EventBridge

Di Amazon EventBridge, Anda dapat mencocokkan peristiwa Secrets Manager dari entri CloudTrail log. Anda dapat mengonfigurasi EventBridge aturan yang mencari peristiwa ini dan kemudian mengirim peristiwa baru yang dihasilkan ke target untuk mengambil tindakan. Untuk daftar CloudTrail entri yang dicatat oleh Secrets Manager, lihatCloudTrail entri. Untuk petunjuk penyiapan EventBridge, lihat Memulai EventBridge di Panduan EventBridge Pengguna.

> **Note**
>
> Topik ini menjelaskan peristiwa Secrets Manager yang cocok yang EventBridge berasal dari entri CloudTrail log. Secrets Manager juga menerbitkan acara asli langsung ke EventBridge, seperti `Secret Label Updated` acara, yang tidak memerlukan CloudTrail. Untuk informasi selengkapnya tentang acara asli, lihatPemberitahuan acara rahasia.

Cocokkan semua perubahan dengan rahasia tertentu

> **Note**
>
> Karena beberapa peristiwa Secrets Manager mengembalikan ARN rahasia dengan kapitalisasi yang berbeda, dalam pola acara yang cocok dengan lebih dari satu tindakan, untuk menentukan rahasia oleh ARN, Anda mungkin perlu menyertakan kunci dan. `arn aRN` Untuk informasi lebih lanjut, lihat AWS re:Post.

Contoh berikut menunjukkan pola EventBridge peristiwa yang cocok dengan entri log untuk perubahan rahasia.

```
{
    "source": ["aws.secretsmanager"],
    "detail-type": ["AWS API Call via CloudTrail"],
    "detail": {
        "eventSource": ["secretsmanager.amazonaws.com"],
        "eventName": ["DeleteResourcePolicy", "PutResourcePolicy", "RotateSecret",
 "TagResource", "UntagResource", "UpdateSecret"],
        "responseElements": {
            "arn": ["arn:aws:secretsmanager:us-west-2:012345678901:secret:mySecret-
a1b2c3"]
        }
    }
}
```

Cocokkan acara saat nilai rahasia berputar

Untuk mendeteksi kapan nilai aktif rahasia berubah, sebaiknya Anda mencocokkan `Secret Label Updated` peristiwa asli untuk label `AWSCURRENT` pementasan. Secrets Manager memindahkan `AWSCURRENT` label ke versi baru setiap kali nilai rahasia aktif berubah, baik dari pembaruan manual atau rotasi otomatis. Acara ini diaktifkan secara default untuk semua rahasia dan dirutekan ke bus EventBridge acara default. Untuk menggunakannya, Anda menambahkan pola acara yang cocok dengannya, seperti yang ditunjukkan pada contoh berikut.

```
{
    "source": ["aws.secretsmanager"],
    "detail-type": ["Secret Label Updated"],
    "detail": {
        "labelUpdated": ["AWSCURRENT"]
    }
}
```

Untuk informasi lebih lanjut tentang `Secret Label Updated` acara tersebut, lihatPemberitahuan acara rahasia.

Atau, Anda dapat mencocokkan entri CloudTrail log untuk operasi yang mengubah nilai rahasia. Beberapa peristiwa ini berasal dari operasi Secrets Manager dan beberapa dihasilkan oleh layanan Secrets Manager. Anda harus menyertakan `detail-type` untuk keduanya. Contoh berikut menunjukkan pola EventBridge peristiwa yang cocok dengan entri CloudTrail log ini.

```
{
    "source": ["aws.secretsmanager"],
    "detail-type": [
        "AWS API Call via CloudTrail",
        "AWS Service Event via CloudTrail"
    ],
    "detail": {
        "eventSource": ["secretsmanager.amazonaws.com"],
        "eventName": ["PutSecretValue", "UpdateSecret", "RotationSucceeded"]
    }
}
```

Pantau kapan AWS Secrets Manager rahasia yang dijadwalkan untuk dihapus diakses

Anda dapat menggunakan kombinasi AWS CloudTrail, Amazon CloudWatch Logs, dan Amazon Simple Notification Service (Amazon SNS) untuk membuat alarm yang memberi tahu Anda tentang upaya apa pun untuk mengakses penghapusan rahasia yang tertunda. Jika Anda menerima pemberitahuan dari alarm, Anda mungkin ingin membatalkan penghapusan rahasia untuk memberi diri Anda lebih banyak waktu untuk menentukan apakah Anda benar-benar ingin menghapusnya. Investigasi Anda mungkin mengakibatkan rahasia dipulihkan karena Anda masih membutuhkan rahasianya. Atau, Anda mungkin perlu memperbarui pengguna dengan rincian rahasia baru untuk digunakan.

Prosedur berikut menjelaskan cara menerima pemberitahuan ketika permintaan untuk `GetSecretValue` operasi yang menghasilkan pesan kesalahan tertentu yang ditulis ke file CloudTrail log Anda. Operasi API lainnya dapat dilakukan secara rahasia tanpa memicu alarm. CloudWatch Alarm ini mendeteksi penggunaan yang mungkin menunjukkan seseorang atau aplikasi menggunakan kredensi yang sudah ketinggalan zaman.

Sebelum memulai prosedur ini, Anda harus mengaktifkan CloudTrail akun Wilayah AWS dan tempat Anda ingin memantau permintaan AWS Secrets Manager API. Untuk instruksi, buka Membuat jejak untuk pertama kalinya di Panduan AWS CloudTrail Pengguna.

Langkah 1: Konfigurasikan pengiriman file CloudTrail log ke CloudWatch Log

Anda harus mengonfigurasi pengiriman file CloudTrail log Anda ke CloudWatch Log. Anda melakukannya agar CloudWatch Log dapat memantau permintaan Secrets Manager API untuk mengambil penghapusan rahasia yang tertunda.

Untuk mengonfigurasi pengiriman file CloudTrail log ke CloudWatch Log

1. Buka CloudTrail konsol di https://console.aws.amazon.com/cloudtrail/.
2. Di bilah navigasi atas, pilih Wilayah AWS untuk memantau rahasia.
3. Di panel navigasi kiri, pilih Jalur, lalu pilih nama jejak yang akan dikonfigurasi. CloudWatch
4. Pada halaman Konfigurasi Jalur, gulir ke bawah ke bagian CloudWatch Log, lalu pilih ikon edit (✏).
5. Untuk grup log baru atau yang sudah ada, ketikkan nama untuk grup log, seperti**`CloudTrail/MyCloudWatchLogGroup`**.
6. Untuk peran IAM, Anda dapat menggunakan peran default bernama CloudTrail_ CloudWatchLogs _Role. Peran ini memiliki kebijakan peran default dengan izin yang diperlukan untuk mengirimkan CloudTrail peristiwa ke grup log.
7. Pilih Lanjutkan untuk menyimpan konfigurasi Anda.
8. Pada saat AWS CloudTrail akan mengirimkan CloudTrail peristiwa yang terkait dengan aktivitas API di akun Anda ke halaman grup CloudWatch log Log Anda, pilih Izinkan.

Langkah 2: Buat CloudWatch alarm

Untuk menerima pemberitahuan saat operasi Secrets Manager `GetSecretValue` API meminta untuk mengakses penghapusan rahasia yang tertunda, Anda harus membuat CloudWatch alarm dan mengonfigurasi notifikasi.

Untuk membuat CloudWatch alarm

1. Masuk ke CloudWatch konsol di https://console.aws.amazon.com/cloudwatch/.
2. Di bilah navigasi atas, pilih AWS Wilayah tempat Anda ingin memantau rahasia.
3. Di panel navigasi bagian kiri, pilih Log.

4. Dalam daftar Grup Log, pilih kotak centang di samping grup log yang Anda buat dalam prosedur sebelumnya, seperti CloudTrail/MyCloudWatchLogGroup. Kemudian pilih Buat Filter Metrik.
5. Untuk Pola Filter, ketik atau tempel yang berikut ini:

   ```
   { $.eventName = "GetSecretValue" && $.errorMessage = "*secret because it was marked
    for deletion*" }
   ```

   Pilih Tetapkan Metrik.
6. Pada halaman Buat Metrik Filter dan Tetapkan Metrik, lakukan hal berikut:

   a. Untuk Namespace Metrik, ketik **`CloudTrailLogMetrics`**.

   b. Untuk Nama Metrik, ketik **`AttemptsToAccessDeletedSecrets`**.

   c. Pilih Tampilkan pengaturan metrik lanjutan, lalu jika perlu untuk Nilai Metrik, ketik**1**.

   d. Pilih Buat Filter.
7. Dalam kotak filter, pilih Buat Alarm.
8. Di jendela Buat Alarm, lakukan hal berikut:

   a. Untuk Nama, ketik **`AttemptsToAccessDeletedSecretsAlarm`**.

   b. Kapanpun:, for is:, pilih **>=**, lalu ketik**1**.

   c. Di samping Kirim pemberitahuan ke:, lakukan salah satu hal berikut:

      - Untuk membuat dan menggunakan topik Amazon SNS baru, pilih Daftar baru, lalu ketik nama topik baru. Untuk Daftar email:, ketik setidaknya satu alamat email. Anda dapat mengetik beberapa alamat email dengan memisahkannya dengan koma.
      - Untuk menggunakan topik Amazon SNS yang sudah ada, pilih nama topik yang akan digunakan. Jika daftar tidak ada, pilih Pilih daftar.

   d. Pilih Buat Alarm.

Langkah 3: Uji CloudWatch alarm

Untuk menguji alarm Anda, buat rahasia dan kemudian jadwalkan untuk dihapus. Kemudian, cobalah untuk mengambil nilai rahasia. Anda segera menerima email di alamat yang Anda konfigurasikan di alarm. Ini mengingatkan Anda untuk penggunaan rahasia yang dijadwalkan untuk dihapus.

Memantau AWS Secrets Manager rahasia untuk kepatuhan dengan menggunakan AWS Config

Anda dapat menggunakan AWS Config untuk mengevaluasi rahasia Anda untuk melihat apakah mereka sesuai dengan standar Anda. Anda menentukan persyaratan keamanan dan kepatuhan internal Anda untuk rahasia menggunakan AWS Config aturan. Kemudian AWS Config dapat mengidentifikasi rahasia yang tidak sesuai dengan aturan Anda. Anda juga dapat melacak perubahan metadata rahasia, konfigurasi rotasi, kunci KMS yang digunakan untuk enkripsi rahasia, fungsi rotasi Lambda, dan tag yang terkait dengan rahasia.

Anda dapat mengonfigurasi AWS Config untuk memberi tahu Anda tentang perubahan. Untuk informasi selengkapnya, lihat Pemberitahuan yang AWS Config mengirim ke topik Amazon SNS.

Jika Anda memiliki rahasia di beberapa Akun AWS dan Wilayah AWS di organisasi Anda, Anda dapat menggabungkan konfigurasi dan data kepatuhan tersebut. Untuk informasi selengkapnya, lihat agregasi Multi-account Multi-Region data.

Untuk menilai apakah rahasia sesuai

- Ikuti petunjuk tentang Mengevaluasi sumber daya Anda dengan AWS Config aturan, dan pilih salah satu aturan berikut:
  - `secretsmanager-secret-unused`— Memeriksa apakah rahasia diakses dalam jumlah hari yang ditentukan.
  - `secretsmanager-using-cmk`— Memeriksa apakah rahasia dienkripsi menggunakan Kunci yang dikelola AWS `aws/secretsmanager` atau kunci yang dikelola pelanggan yang Anda buat. AWS KMS
  - `secretsmanager-rotation-enabled-check`— Memeriksa apakah rotasi dikonfigurasi untuk rahasia yang disimpan di Secrets Manager.
  - `secretsmanager-scheduled-rotation-success-check`— Memeriksa apakah rotasi sukses terakhir berada dalam frekuensi rotasi yang dikonfigurasi. Frekuensi minimum untuk cek adalah setiap hari.
  - `secretsmanager-secret-periodic-rotation`— Memeriksa apakah rahasia diputar dalam jumlah hari yang ditentukan.

Biaya Monitor Secrets Manager

Anda dapat menggunakan Amazon CloudWatch untuk memantau perkiraan AWS Secrets Manager biaya. Untuk informasi selengkapnya, lihat Membuat alarm penagihan untuk memantau perkiraan AWS tagihan Anda di Panduan CloudWatch Pengguna.

Pilihan lain untuk memantau biaya Anda adalah Deteksi Anomali AWS Biaya. Untuk informasi selengkapnya, lihat Mendeteksi pengeluaran yang tidak biasa dengan Deteksi Anomali AWS Biaya di Panduan Pengguna Manajemen AWS Biaya.

Untuk informasi tentang memantau penggunaan Secrets Manager Anda, lihat the section called "Monitor dengan CloudWatch" danthe section called "Masuk dengan AWS CloudTrail ".

Untuk informasi tentang AWS Secrets Manager harga, lihatthe section called "Harga".

Mendeteksi ancaman dengan Amazon GuardDuty

Amazon GuardDuty adalah layanan deteksi ancaman yang membantu Anda melindungi akun, wadah, beban kerja, dan data dengan AWS lingkungan Anda. Dengan menggunakan model machine learning (ML) dan kemampuan deteksi anomali dan ancaman, GuardDuty terus memantau sumber log yang berbeda untuk mengidentifikasi dan memprioritaskan potensi risiko keamanan dan aktivitas berbahaya di lingkungan Anda. Misalnya, GuardDuty akan mendeteksi potensi ancaman seperti akses yang tidak biasa atau mencurigakan ke rahasia, dan eksfiltrasi kredenal jika mendeteksi kredenal yang dibuat secara eksklusif untuk instans Amazon EC2 melalui peran peluncuran instans tetapi digunakan dari akun lain di dalamnya. AWS Untuk informasi selengkapnya, lihat Panduan GuardDuty Pengguna Amazon.

Contoh lain kasus penggunaan untuk deteksi adalah perilaku anomali. Misalnya, jika AWS Secrets Manager biasanya mendapat`create-secret`, `get-secret-value``describe-secret`, dan `list-secrets` panggilan dari entitas yang menggunakan Java SDK, dan kemudian entitas yang berbeda mulai memanggil `batch-get-secret-value` dan `get-secret-value` menggunakan AWS CLI dari luar VPN, GuardDuty dapat melaporkan temuan bahwa entitas kedua secara anomali menjalankan API. Untuk informasi selengkapnya, lihat tipe CredentialAccess:IAMUser/AnomalousBehavior pencarian GuardDuty IAM.

Validasi kepatuhan untuk AWS Secrets Manager

Tanggung jawab kepatuhan Anda saat menggunakan Secrets Manager ditentukan oleh sensitivitas data Anda, tujuan kepatuhan perusahaan Anda, serta undang-undang dan peraturan yang berlaku. AWS menyediakan sumber daya berikut untuk membantu kepatuhan:

- Panduan Quick Start Keamanan dan Kepatuhan – Panduan deployment ini membahas pertimbangan arsitektur dan menyediakan langkah–langkah untuk melakukan deployment terhadap lingkungan dasar di AWS yang menjadi fokus keamanan dan kepatuhan.
- Arsitektur untuk HIPAA Security and Compliance Whitepaper — Whitepaper ini menjelaskan bagaimana perusahaan dapat menggunakan untuk membuat aplikasi. AWS HIPAA-compliant
- AWS Sumber Daya AWS — Kumpulan buku kerja dan panduan ini mungkin berlaku untuk industri dan lokasi Anda.
- AWS Configmenilai seberapa baik konfigurasi sumber daya Anda mematuhi praktik internal, pedoman industri, dan peraturan. Untuk informasi selengkapnya, lihat the section called “Pantau rahasia untuk kepatuhan”.
- AWS Security Hub CSPMmemberikan pandangan komprehensif tentang keadaan keamanan Anda di dalamnya AWS yang membantu Anda memeriksa kepatuhan Anda terhadap standar industri keamanan dan praktik terbaik. Untuk informasi tentang menggunakan Security Hub CSPM untuk mengevaluasi sumber daya Secrets Manager, lihat AWS Secrets Manager kontrol di AWS Security Hub CSPM Panduan Pengguna.
- IAM Access Analyzer menganalisis kebijakan, termasuk pernyataan kondisi dalam kebijakan, yang memungkinkan entitas eksternal mengakses rahasia. Untuk informasi selengkapnya, lihat Mempratinjau akses dengan Access Analyzer.
- AWS Systems Managermenyediakan runbook standar untuk Secrets Manager. Untuk informasi selengkapnya, lihat referensi runbook Automation Systems Manager untuk Secrets Manager.
- Anda dapat mengunduh laporan audit pihak ketiga menggunakan AWS Artifact. Untuk informasi selengkapnya, lihat Mengunduh Laporan di AWS Artifact .

Standar kepatuhan

AWS Secrets Manager telah menjalani audit untuk standar berikut dan dapat menjadi bagian dari solusi Anda ketika Anda perlu mendapatkan sertifikasi kepatuhan.

- HIPAA — AWS telah memperluas program kepatuhan Health Insurance Portability and Accountability Act (HIPAA) untuk dimasukkan sebagai layanan. AWS Secrets Manager HIPAA-eligible Jika Anda memiliki Business Associate Agreement (BAA) yang dieksekusi AWS, Anda dapat menggunakan Secrets Manager untuk membantu membangun HIPAA-compliant aplikasi Anda. AWS menawarkan HIPAA-focused whitepaper untuk pelanggan yang tertarik AWS untuk mempelajari lebih lanjut tentang bagaimana mereka dapat memanfaatkan pemrosesan dan penyimpanan informasi kesehatan. Untuk informasi lebih lanjut, lihat Kepatuhan HIPAA.
- Organisasi Partisipasi PCI — AWS Secrets Manager memiliki Pengesahan Kepatuhan untuk Standar Keamanan Data (DSS) Industri Kartu Pembayaran (PCI) versi 3.2 di Penyedia Layanan Level 1. Pelanggan yang menggunakan AWS produk dan layanan untuk menyimpan, memproses, atau mengirimkan data pemegang kartu dapat menggunakannya AWS Secrets Manager saat mereka mengelola sertifikasi kepatuhan PCI DSS mereka sendiri. Untuk informasi selengkapnya tentang PCI DSS, termasuk cara meminta salinan PCI AWS Compliance Package, lihat PCI DSS Level 1.
- ISO - AWS Secrets Manager telah berhasil menyelesaikan sertifikasi kepatuhan untuk ISO/IEC 27001, ISO/IEC 27017, ISO/IEC 27018, dan ISO 9001. Untuk informasi selengkapnya, lihat ISO 27001, ISO 27017, ISO 27018, ISO 9001.
- Laporan AICPA SOC — Sistem dan Kontrol Organisasi (SOC) adalah laporan pemeriksaan pihak ketiga independen yang menunjukkan bagaimana Secrets Manager mencapai kontrol dan tujuan kepatuhan utama. Tujuan dari laporan ini adalah untuk membantu Anda dan auditor Anda memahami AWS kontrol yang ditetapkan untuk mendukung operasi dan kepatuhan. Untuk informasi selengkapnya, lihat Kepatuhan SOC.
- FedRAMP — Federal Risk and Authorization Management Program (FedRAMP) adalah program pemerintah yang menyediakan pendekatan standar untuk penilaian keamanan, otorisasi, dan pemantauan berkelanjutan untuk produk dan layanan cloud. Program FedRAMP juga menyediakan otorisasi sementara untuk layanan dan wilayah East/West untuk dan untuk mengkonsumsi data pemerintah atau yang GovCloud diatur. Untuk informasi lain, lihat Kepatuhan FedRAMP .
- Departemen Pertahanan — The Department of Defense (DoD) Cloud Computing Security Requirements Guide (SRG) menyediakan penilaian standar dan proses otorisasi untuk penyedia layanan cloud (CSP) untuk mendapatkan otorisasi sementara DoD, sehingga mereka dapat melayani pelanggan DoD. Untuk informasi selengkapnya, lihat DoD SRG Resources
- IRAP — Information Security Registered Assesors Program (IRAP) memungkinkan pelanggan pemerintah Australia untuk memvalidasi bahwa kontrol yang tepat telah ada dan menentukan model tanggung jawab yang sesuai untuk menangani persyaratan Manual Keamanan Informasi

pemerintah Australia (ISM) yang diproduksi oleh Australian Cyber Security Centre (ACSC). Untuk informasi lebih lanjut, lihat Sumber Daya IRAP

- OSPAR — Amazon Web Services (AWS) mencapai pengesahan Laporan Audit Penyedia Layanan Outsourced (OSPAR). AWS Keselarasan dengan Pedoman Asosiasi Bank di Singapura (ABS) tentang Tujuan Pengendalian dan Prosedur untuk Penyedia Layanan Outsourced (Pedoman ABS) menunjukkan kepada pelanggan AWS komitmen untuk memenuhi harapan tinggi bagi penyedia layanan cloud yang ditetapkan oleh industri jasa keuangan di Singapura. Untuk informasi selengkapnya, lihat Sumber Daya OSPAR

Keamanan di AWS Secrets Manager

Keamanan 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.

Anda dan AWS berbagi tanggung jawab untuk keamanan. Model tanggung jawab bersama menggambarkan ini sebagai keamanan cloud dan keamanan di 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. Third-party Auditor secara teratur menguji dan memverifikasi efektivitas keamanan kami sebagai bagian dari Program AWS Kepatuhan. Untuk mempelajari tentang program kepatuhan yang berlaku AWS Secrets Manager, lihat AWS Layanan dalam Lingkup berdasarkan Program Kepatuhan.
- Keamanan di cloud — AWS Layanan Anda menentukan tanggung jawab Anda. Anda juga bertanggung jawab atas faktor lain, yang mencakup sensitivitas data Anda, persyaratan perusahaan Anda, serta undang-undang dan peraturan yang berlaku.

Untuk sumber daya lainnya, lihat Pilar Keamanan — AWS Well-Architected Kerangka Kerja.

Topik

- Mengurangi risiko menggunakan AWS CLI untuk menyimpan rahasia Anda AWS Secrets Manager
- Kendali autentikasi dan akses untuk AWS Secrets Manager
- Perlindungan data di AWS Secrets Manager
- Enkripsi rahasia dan dekripsi di AWS Secrets Manager
- Keamanan infrastruktur di AWS Secrets Manager
- Menggunakan titik akhir AWS Secrets Manager VPC
- Kontrol akses API dengan kebijakan IAM
- Ketahanan di AWS Secrets Manager
- Post-quantum TLS

Mengurangi risiko menggunakan AWS CLI untuk menyimpan rahasia Anda AWS Secrets Manager

Saat Anda menggunakan AWS Command Line Interface (AWS CLI) untuk menjalankan AWS operasi, Anda memasukkan perintah tersebut di shell perintah. Misalnya, Anda dapat menggunakan prompt perintah Windows atau Windows PowerShell, atau shell Bash atau Z, antara lain. Banyak dari shell perintah ini mencakup fungsionalitas yang dirancang untuk meningkatkan produktivitas. Tetapi fungsi ini dapat digunakan untuk membahayakan rahasia Anda. Misalnya, di sebagian besar shell, Anda dapat menggunakan tombol panah atas untuk melihat perintah yang terakhir dimasukkan. Fitur riwayat perintah dapat dimanfaatkan oleh siapa saja yang mengakses sesi tanpa jaminan Anda. Selain itu, utilitas lain yang bekerja di latar belakang mungkin memiliki akses ke parameter perintah Anda, dengan tujuan yang dimaksudkan untuk membantu Anda melakukan tugas dengan lebih efisien. Untuk mengurangi risiko tersebut, pastikan Anda mengambil langkah-langkah berikut:

- Selalu kunci komputer Anda ketika Anda berjalan menjauh dari konsol Anda.
- Copot pemasangan atau nonaktifkan utilitas konsol yang tidak perlu atau tidak lagi digunakan.
- Pastikan shell atau program akses jarak jauh, jika Anda menggunakan salah satu atau yang lain, jangan mencatat perintah yang diketik.
- Gunakan teknik untuk meneruskan parameter yang tidak ditangkap oleh riwayat perintah shell. Contoh berikut menunjukkan bagaimana Anda dapat mengetik teks rahasia ke dalam file teks, dan kemudian meneruskan file ke AWS Secrets Manager perintah dan segera menghancurkan file. Ini berarti riwayat shell tipikal tidak menangkap teks rahasia.

  Contoh berikut menunjukkan perintah Linux yang khas tetapi shell Anda mungkin memerlukan perintah yang sedikit berbeda:

```
$ touch secret.txt
        # Creates an empty text file
$ chmod go-rx secret.txt
        # Restricts access to the file to only the user
$ cat > secret.txt
        # Redirects standard input (STDIN) to the text file
ThisIsMyTopSecretPassword^D
        # Everything the user types from this point up to the CTRL-D (^D) is saved in
 the file
$ aws secretsmanager create-secret --name TestSecret --secret-string file://
secret.txt       # The Secrets Manager command takes the --secret-string parameter
 from the contents of the file
```

```
$ shred -u secret.txt
        # The file is destroyed so it can no longer be accessed.
```

Setelah Anda menjalankan perintah ini, Anda harus dapat menggunakan panah atas dan bawah untuk menggulir melalui riwayat perintah dan melihat bahwa teks rahasia tidak ditampilkan pada baris apa pun.

> **Important**
>
> Secara default, Anda tidak dapat melakukan teknik yang setara di Windows kecuali Anda terlebih dahulu mengurangi ukuran buffer riwayat perintah menjadi 1.

Untuk mengkonfigurasi Windows Command Prompt untuk hanya memiliki 1 buffer riwayat perintah dari 1 perintah

1. Buka prompt perintah Administrator (Jalankan sebagai administrator).
2. Pilih ikon di kiri atas dan kemudian pilih Properties.
3. Pada tab Opsi, atur Ukuran Buffer dan Jumlah Buffer keduanya ke**1**, lalu pilih OK.
4. Setiap kali Anda harus mengetik perintah yang tidak Anda inginkan dalam riwayat, segera ikuti dengan satu perintah lain, seperti:

   ```
   echo.
   ```

   Ini memastikan Anda menyiram perintah sensitif.

Untuk shell Windows Command Prompt, Anda dapat mengunduh alat SysInternalsSDelete, dan kemudian menggunakan perintah yang mirip dengan berikut ini:

```
C:\> echo. 2> secret.txt
        # Creates an empty file
C:\> icacls secret.txt /remove "BUILTIN\Administrators" "NT AUTHORITY/SYSTEM" /
inheritance:r   # Restricts access to the file to only the owner
C:\> copy con secret.txt /y
        # Redirects the keyboard to text file, suppressing prompt to overwrite
THIS IS MY TOP SECRET PASSWORD^Z
      # Everything the user types from this point up to the CTRL-Z (^Z) is saved in the
 file
```

```
C:\> aws secretsmanager create-secret --name TestSecret --secret-string file://
secret.txt      # The Secrets Manager command takes the --secret-string parameter from
 the contents of the file
C:\> sdelete secret.txt
        # The file is destroyed so it can no longer be accessed.
```

Kendali autentikasi dan akses untuk AWS Secrets Manager

Secrets Manager menggunakan AWS Identity and Access Management (IAM) untuk mengamankan akses ke rahasia. IAM menyediakan otentikasi dan kontrol akses. Otentikasi memverifikasi identitas permintaan individu. Secrets Manager menggunakan proses masuk dengan kata sandi, kunci akses, dan token otentikasi multi-faktor (MFA) untuk memverifikasi identitas pengguna. Lihat Masuk ke AWS. Kontrol akses memastikan bahwa hanya individu yang disetujui yang dapat melakukan operasi pada AWS sumber daya seperti rahasia. Secrets Manager menggunakan kebijakan untuk menentukan siapa yang memiliki akses ke sumber daya mana, dan tindakan apa yang dapat diambil identitas terhadap sumber daya tersebut. Lihat Kebijakan dan izin di IAM.

Topik

- Referensi izin untuk AWS Secrets Manager
- Izin administrator Secrets Manager
- Izin untuk mengakses rahasia
- Izin untuk fungsi rotasi Lambda
- Izin untuk kunci enkripsi
- Izin untuk replikasi
- Identity-based kebijakan
- Resource-based kebijakan
- Kontrol akses ke rahasia menggunakan kontrol akses berbasis atribut (ABAC)
- AWS kebijakan yang dikelola untuk AWS Secrets Manager
- Tentukan siapa yang memiliki izin untuk Anda AWS Secrets Manager rahasia
- Akses AWS Secrets Manager rahasia dari akun yang berbeda
- Mengakses rahasia dari lingkungan lokal

Referensi izin untuk AWS Secrets Manager

Referensi izin untuk Secrets Manager tersedia di Actions, resources, dan condition key untuk AWS Secrets Manager Referensi Otorisasi Layanan.

Izin administrator Secrets Manager

Untuk memberikan izin administrator Secrets Manager, ikuti petunjuk di Menambahkan dan menghapus izin identitas IAM, dan lampirkan kebijakan berikut:

- SecretsManagerReadWrite
- IAMFullAccess

Kami menyarankan Anda untuk tidak memberikan izin administrator kepada pengguna akhir. Meskipun ini memungkinkan pengguna Anda untuk membuat dan mengelola rahasia mereka, izin yang diperlukan untuk mengaktifkan rotation (IAMFullAccess) memberikan izin signifikan yang tidak sesuai untuk pengguna akhir.

Izin untuk mengakses rahasia

Dengan menggunakan kebijakan izin IAM, Anda mengontrol pengguna atau layanan mana yang memiliki akses ke rahasia Anda. Kebijakan izin menjelaskan siapa yang dapat melakukan tindakan apa pada sumber daya mana. Anda dapat:

- the section called “Identity-based kebijakan”
- the section called “Resource-based kebijakan”

Izin untuk fungsi rotasi Lambda

Secrets Manager menggunakan AWS Lambda fungsi untuk memutar rahasia. Fungsi Lambda harus memiliki akses ke rahasia serta database atau layanan yang rahasia berisi kredensialnya. Lihat Izin untuk rotasi.

Izin untuk kunci enkripsi

Secrets Manager menggunakan AWS Key Management Service (AWS KMS) kunci untuk mengenkripsi rahasia. Kunci yang dikelola AWS `aws/secretsmanager`Secara otomatis memiliki

izin yang benar. Jika Anda menggunakan kunci KMS yang berbeda, Secrets Manager memerlukan izin untuk kunci tersebut. Lihat the section called “Izin untuk kunci KMS”.

Izin untuk replikasi

Dengan menggunakan kebijakan izin IAM, Anda mengontrol pengguna atau layanan mana yang dapat mereplikasi rahasia Anda ke Wilayah lain. Lihat the section called “Mencegah replikasi”.

Identity-based kebijakan

Anda dapat melampirkan kebijakan izin ke identitas IAM: pengguna, grup pengguna, dan peran. Dalam kebijakan berbasis identitas, Anda menentukan rahasia mana yang dapat diakses identitas dan tindakan yang dapat dilakukan identitas pada rahasia. Untuk informasi selengkapnya, lihat Menambahkan dan menghapus izin identitas IAM.

Anda dapat memberikan izin untuk peran yang mewakili aplikasi atau pengguna di layanan lain. Misalnya, aplikasi yang berjalan pada instans Amazon EC2 mungkin memerlukan akses ke database. Anda dapat membuat peran IAM yang dilampirkan ke profil instans EC2 dan kemudian menggunakan kebijakan izin untuk memberikan akses peran ke rahasia yang berisi kredensil untuk database. Untuk informasi selengkapnya, lihat Menggunakan peran IAM untuk memberikan izin ke aplikasi yang berjalan di instans Amazon EC2. Layanan lain yang dapat Anda lampirkan peran untuk menyertakan Amazon Redshift, AWS Lambda, dan Amazon ECS.

Anda juga dapat memberikan izin kepada pengguna yang diautentikasi oleh sistem identitas selain IAM. Misalnya, Anda dapat mengaitkan peran IAM ke pengguna aplikasi seluler yang masuk dengan Amazon Cognito. Peran tersebut memberikan kredensi sementara aplikasi dengan izin dalam kebijakan izin peran. Kemudian Anda dapat menggunakan kebijakan izin untuk memberikan akses peran ke rahasia. Untuk informasi selengkapnya, lihat Penyedia identitas dan federasi.

Anda dapat menggunakan kebijakan berbasis identitas untuk:

- Berikan akses identitas ke beberapa rahasia.
- Kontrol siapa yang dapat membuat rahasia baru, dan siapa yang dapat mengakses rahasia yang belum dibuat.
- Berikan akses grup IAM ke rahasia.

Contoh:

- Contoh: Izin untuk mengambil nilai rahasia individu

- Contoh: Izin untuk membaca dan menggambarkan rahasia individu
- Contoh: Izin untuk mengambil sekelompok nilai rahasia dalam batch
- Contoh: Wildcard
- Contoh: Izin untuk membuat rahasia
- Contoh: Tolak AWS KMS kunci tertentu untuk mengenkripsi rahasia

Contoh: Izin untuk mengambil nilai rahasia individu

Untuk memberikan izin untuk mengambil nilai rahasia, Anda dapat melampirkan kebijakan ke rahasia atau identitas. Untuk bantuan menentukan jenis kebijakan yang akan digunakan, lihat Identity-basedkebijakan dan kebijakan berbasis sumber daya. Untuk informasi tentang cara melampirkan kebijakan, lihat the section called “Resource-based kebijakan” danthe section called “Identity-based kebijakan”.

Contoh ini berguna ketika Anda ingin memberikan akses ke grup IAM. Untuk memberikan izin untuk mengambil sekelompok rahasia dalam panggilan API batch, lihatthe section called “Contoh: Izin untuk mengambil sekelompok nilai rahasia dalam batch”.

Example Baca rahasia yang dienkripsi menggunakan kunci yang dikelola pelanggan

Jika rahasia dienkripsi menggunakan kunci yang dikelola pelanggan, Anda dapat memberikan akses untuk membaca rahasia dengan melampirkan kebijakan berikut ke identitas. \

JSON

```
{
"Version":"2012-10-17",
"Statement": [
{
  "Effect": "Allow",
  "Action": "secretsmanager:GetSecretValue",
  "Resource": "arn:aws:secretsmanager:us-east-1:123456789012:secret:secretName-
AbCdEf"
},
{
  "Effect": "Allow",
  "Action": "kms:Decrypt",
  "Resource": "arn:aws:kms:us-east-1:123456789012:key/key-id"
```

```
}
]
}
```

Contoh: Izin untuk membaca dan menggambarkan rahasia individu

Example Baca dan jelaskan satu rahasia

Anda dapat memberikan akses ke rahasia dengan melampirkan kebijakan berikut ke identitas.

JSON

```
{
"Version":"2012-10-17",
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "secretsmanager:GetSecretValue",
      "secretsmanager:DescribeSecret"
      ],
      "Resource": "arn:aws:secretsmanager:us-
east-1:123456789012:secret:secretName-AbCdEf"
    }
]
}
```

Contoh: Izin untuk mengambil sekelompok nilai rahasia dalam batch

Example Baca sekelompok rahasia dalam satu batch

Anda dapat memberikan akses untuk mengambil grup rahasia dalam panggilan API batch dengan melampirkan kebijakan berikut ke identitas. Kebijakan membatasi pemanggil sehingga mereka hanya dapat mengambil rahasia yang ditentukan oleh*SecretARN1*,, dan *SecretARN2SecretARN3*, bahkan jika panggilan batch menyertakan rahasia lain. Jika penelepon juga meminta rahasia lain dalam panggilan API batch, Secrets Manager tidak akan mengembalikannya. Untuk informasi lebih lanjut, lihat `BatchGetSecretValue.` .

JSON

```
{
"Version":"2012-10-17",
"Statement": [
{
  "Effect": "Allow",
  "Action": [
    "secretsmanager:BatchGetSecretValue",
    "secretsmanager:ListSecrets"
  ],
  "Resource": "*"
},
{
  "Effect": "Allow",
  "Action": [
    "secretsmanager:GetSecretValue"
  ],
  "Resource": [
    "arn:aws:secretsmanager:us-east-1:123456789012:secret:secretName1-AbCdEf",
    "arn:aws:secretsmanager:us-east-1:123456789012:secret:secretName2-AbCdEf",
    "arn:aws:secretsmanager:us-east-1:123456789012:secret:secretName3-AbCdEf"
    ]
}
]
}
```

Contoh: Wildcard

Anda dapat menggunakan wildcard untuk menyertakan sekumpulan nilai dalam elemen kebijakan.

Example Akses semua rahasia di jalur

Kebijakan berikut memberikan akses untuk mengambil semua rahasia dengan nama yang diawali dengan ""*TestEnv/*.

JSON

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

```
    "Statement": {
        "Effect": "Allow",
        "Action": "secretsmanager:GetSecretValue",
        "Resource": "arn:aws:secretsmanager:us-
east-1:123456789012:secret:TestEnv/*"
    }
}
```

Example Akses metadata pada semua rahasia

Pemberian `DescribeSecret` dan izin kebijakan berikut dimulai dengan`List`: `ListSecrets` dan. `ListSecretVersionIds`

JSON

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

Example Cocokkan nama rahasia

Kebijakan berikut memberikan semua izin Secrets Manager untuk rahasia berdasarkan nama. Untuk menggunakan kebijakan ini, lihatthe section called “Identity-based kebijakan”.

Untuk mencocokkan nama rahasia, Anda membuat ARN untuk rahasia dengan menyusun Region, ID Akun, nama rahasia, dan wildcard (?) untuk mencocokkan karakter acak individual. Secrets Manager menambahkan enam karakter acak ke nama rahasia sebagai bagian dari ARN mereka, sehingga Anda dapat menggunakan wildcard ini untuk mencocokkan karakter tersebut. Jika Anda menggunakan sintaks`"another_secret_name-*"`, Secrets Manager tidak hanya cocok dengan rahasia yang dimaksudkan dengan 6 karakter acak, tetapi juga cocok`"another_secret_name-<anything-here>a1b2c3"`.

Karena Anda dapat memprediksi semua bagian ARN rahasia kecuali 6 karakter acak, menggunakan `'??????'` sintaks karakter wildcard memungkinkan Anda memberikan izin dengan aman ke rahasia yang belum ada. Namun, ketahuilah, jika Anda menghapus rahasia dan membuatnya kembali dengan nama yang sama, pengguna secara otomatis menerima izin untuk rahasia baru, meskipun 6 karakter berubah.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "secretsmanager:*",
            "Resource": [
                "arn:aws:secretsmanager:us-
east-1:123456789012:secret:a_specific_secret_name-a1b2c3",
                "arn:aws:secretsmanager:us-
east-1:123456789012:secret:another_secret_name-??????"
            ]
        }
    ]
}
```

Contoh: Izin untuk membuat rahasia

Untuk memberikan izin pengguna untuk membuat rahasia, kami sarankan Anda melampirkan kebijakan izin ke grup IAM milik pengguna. Lihat grup pengguna IAM.

Example Buat rahasia

Kebijakan berikut memberikan izin untuk membuat rahasia dan melihat daftar rahasia. Untuk menggunakan kebijakan ini, lihatthe section called “Identity-based kebijakan”.

JSON

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

```
  "Effect": "Allow",
  "Action": [
    "secretsmanager:CreateSecret",
    "secretsmanager:ListSecrets"
  ],
  "Resource": "*"
}
]
}
```

Contoh: Tolak AWS KMS kunci tertentu untuk mengenkripsi rahasia

> **Important**
>
> Untuk menolak kunci yang dikelola pelanggan, kami sarankan Anda membatasi akses menggunakan kebijakan kunci atau hibah kunci. Untuk informasi selengkapnya, lihat Otentikasi dan kontrol akses AWS KMS di Panduan AWS Key Management Service Pengembang.

Example Tolak kunci yang AWS dikelola **`aws/secretsmanager`**

Kebijakan berikut menyangkal penggunaan Kunci yang dikelola AWS `aws/secretsmanager` untuk membuat atau memperbarui rahasia. Kebijakan ini memerlukan rahasia untuk dienkripsi menggunakan kunci yang dikelola pelanggan. Kebijakan ini mencakup dua pernyataan:

1. Pernyataan pertama,`Sid: "RequireCustomerManagedKeysOnSecrets"`, menyangkal permintaan untuk membuat atau memperbarui rahasia menggunakan file. Kunci yang dikelola AWS `aws/secretsmanager`
2. Pernyataan kedua,`Sid: "RequireKmsKeyIdParameterOnCreate"`, menolak permintaan untuk membuat rahasia yang tidak menyertakan kunci KMS, karena Secrets Manager akan default menggunakan file. Kunci yang dikelola AWS `aws/secretsmanager`

JSON

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

```
  {
      "Sid": "RequireCustomerManagedKeysOnSecrets",
      "Effect": "Deny",
      "Action": [
          "secretsmanager:CreateSecret",
          "secretsmanager:UpdateSecret"
      ],
      "Resource": "*",
      "Condition": {
          "StringLikeIfExists": {
              "secretsmanager:KmsKeyArn": "<key_ARN_of_the_AWS_managed_key>"
          }
      }
  },
  {
      "Sid": "RequireKmsKeyIdParameterOnCreate",
      "Effect": "Deny",
      "Action": "secretsmanager:CreateSecret",
      "Resource": "*",
      "Condition": {
          "Null": {
              "secretsmanager:KmsKeyArn": "true"
          }
      }
  }
]
}
```

Resource-based kebijakan

Dalam kebijakan berbasis sumber daya, Anda menentukan siapa yang dapat mengakses rahasia dan tindakan yang dapat mereka lakukan pada rahasia tersebut. Anda dapat menggunakan kebijakan berbasis sumber daya untuk:

- Berikan akses ke satu rahasia untuk beberapa pengguna dan peran.
- Berikan akses ke pengguna atau peran di AWS akun lain.

Saat Anda melampirkan kebijakan berbasis sumber daya ke rahasia di konsol, Manajer Rahasia menggunakan mesin penalaran otomatis Zelkova dan API `ValidateResourcePolicy` untuk mencegah Anda memberikan akses kepada berbagai kepala sekolah IAM ke rahasia Anda. Atau,

Anda dapat memanggil `PutResourcePolicy` API dengan `BlockPublicPolicy` parameter dari CLI atau SDK.

> **Important**
>
> Validasi kebijakan sumber daya dan `BlockPublicPolicy` parameter membantu melindungi sumber daya Anda dengan mencegah akses publik diberikan melalui kebijakan sumber daya yang dilampirkan langsung ke rahasia Anda. Selain menggunakan fitur-fitur ini, periksa dengan cermat kebijakan berikut untuk mengonfirmasi bahwa mereka tidak memberikan akses publik:
>
> - Identity-based kebijakan yang dilampirkan ke AWS prinsipal terkait (misalnya, peran IAM)
> - Resource-based kebijakan yang dilampirkan ke AWS sumber daya terkait (misalnya, kunci AWS Key Management Service (AWS KMS))
>
> Untuk meninjau izin ke rahasia Anda, lihatTentukan siapa yang memiliki izin untuk rahasia Anda.

Untuk melihat, mengubah, atau menghapus kebijakan sumber daya untuk rahasia (konsol)

1. Buka konsol Secrets Manager di https://console.aws.amazon.com/secretsmanager/.
2. Dari daftar rahasia, pilih rahasia Anda.
3. Pada halaman detail rahasia, pada tab Ikh tisar, di bagian Izin sumber daya, pilih Edit izin.
4. Di bidang kode, lakukan salah satu hal berikut, lalu pilih S impan:
   - Untuk melampirkan atau memodifikasi kebijakan sumber daya, masukkan kebijakan tersebut.
   - Untuk menghapus kebijakan, kosongkan bidang kode.

AWS CLI

Example Mengambil kebijakan sumber daya

`get-resource-policy`Contoh berikut mengambil kebijakan berbasis sumber daya yang dilampirkan ke rahasia.

```
aws secretsmanager get-resource-policy \
```

```
    --secret-id MyTestSecret
```

Example Menghapus kebijakan sumber daya

`delete-resource-policy`Contoh berikut menghapus kebijakan berbasis sumber daya yang dilampirkan pada rahasia.

```
aws secretsmanager delete-resource-policy \
    --secret-id MyTestSecret
```

Example Menambahkan kebijakan sumber daya

`put-resource-policy`Contoh berikut menambahkan kebijakan izin ke rahasia, memeriksa terlebih dahulu bahwa kebijakan tersebut tidak menyediakan akses luas ke rahasia tersebut. Kebijakan dibaca dari file. Untuk informasi selengkapnya, lihat AWS CLI Memuat parameter dari file di Panduan AWS CLI Pengguna.

```
aws secretsmanager put-resource-policy \
    --secret-id MyTestSecret \
    --resource-policy file://mypolicy.json \
    --block-public-policy
```

Isi dari `mypolicy.json`:

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::123456789012:role/MyRole"
            },
            "Action": "secretsmanager:GetSecretValue",
            "Resource": "*"
        }
    ]
}
```

AWS SDK

Untuk mengambil kebijakan yang dilampirkan ke rahasia, gunakan `GetResourcePolicy`.

Untuk menghapus kebijakan yang dilampirkan ke rahasia, gunakan `DeleteResourcePolicy`.

Untuk melampirkan kebijakan ke rahasia, gunakan `PutResourcePolicy`. Jika sudah ada kebijakan yang dilampirkan, perintah akan menggantinya dengan kebijakan baru. Kebijakan harus diformat sebagai teks terstruktur JSON. Lihat Struktur dokumen kebijakan JSON.

Untuk informasi selengkapnya, lihat the section called “AWS SDK”.

Contoh

Contoh:

- Contoh: Izin untuk mengambil nilai rahasia individu
- Contoh: Izin dan VPC
- Contoh: Prinsip layanan

Contoh: Izin untuk mengambil nilai rahasia individu

Untuk memberikan izin untuk mengambil nilai rahasia, Anda dapat melampirkan kebijakan ke rahasia atau identitas. Untuk bantuan menentukan jenis kebijakan yang akan digunakan, lihat Identity-based kebijakan dan kebijakan berbasis sumber daya. Untuk informasi tentang cara melampirkan kebijakan, lihat the section called “Resource-based kebijakan” danthe section called “Identity-based kebijakan”.

Contoh ini berguna ketika Anda ingin memberikan akses ke satu rahasia untuk beberapa pengguna atau peran. Untuk memberikan izin untuk mengambil sekelompok rahasia dalam panggilan API batch, lihatthe section called “Contoh: Izin untuk mengambil sekelompok nilai rahasia dalam batch”.

Example Baca satu rahasia

Anda dapat memberikan akses ke rahasia dengan melampirkan kebijakan berikut ke rahasia.

JSON

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

```
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:role/EC2RoleToAccessSecrets"
            },
            "Action": "secretsmanager:GetSecretValue",
            "Resource": "*"
        }
    ]
}
```

Contoh: Izin dan VPC

Jika Anda perlu mengakses Manajer Rahasia dari dalam VPC, Anda dapat memastikan bahwa permintaan ke Manajer Rahasia berasal dari VPC dengan menyertakan kondisi dalam kebijakan izin Anda. Untuk informasi selengkapnya, lihat Batasi permintaan dengan kondisi titik akhir VPC dan the section called “Titik akhir VPC (AWS PrivateLink)”.

Pastikan bahwa permintaan untuk mengakses rahasia dari AWS layanan lain juga berasal dari VPC, jika tidak kebijakan ini akan menolak akses mereka.

Example Memerlukan permintaan untuk datang melalui titik akhir VPC

Kebijakan berikut memungkinkan pengguna untuk melakukan operasi Manajer Rahasia hanya ketika permintaan datang melalui titik akhir VPC. *vpce-1234a5678b9012c*

JSON

```
{
"Id": "example-policy-1",
"Version":"2012-10-17",
"Statement": [
{
  "Sid": "RestrictGetSecretValueoperation",
  "Effect": "Deny",
  "Principal": "*",
  "Action": "secretsmanager:GetSecretValue",
  "Resource": "*",
  "Condition": {
    "StringNotEquals": {
      "aws:sourceVpce": "vpce-12345678"
```

```
    }
  }
}
]
}
```

Example Memerlukan permintaan untuk datang dari VPC

Kebijakan berikut mengizinkan perintah untuk membuat dan mengelola rahasia hanya jika berasal dari*vpc-12345678*. Selain itu, kebijakan ini memungkinkan operasi yang menggunakan akses nilai terenkripsi rahasia hanya ketika permintaan berasalvpc-2b2b2b2b. Anda dapat menggunakan kebijakan seperti ini jika menjalankan aplikasi dalam satu VPC, tetapi Anda menggunakan VPC kedua yang terisolasi untuk fungsi manajemen.

JSON

```
{
"Id": "example-policy-2",
"Version":"2012-10-17",
"Statement": [
{
  "Sid": "AllowAdministrativeActionsfromONLYvpc-12345678",
  "Effect": "Deny",
  "Principal": "*",
  "Action": [
    "secretsmanager:Create*",
    "secretsmanager:Put*",
    "secretsmanager:Update*",
    "secretsmanager:Delete*",
    "secretsmanager:Restore*",
    "secretsmanager:RotateSecret",
    "secretsmanager:CancelRotate*",
    "secretsmanager:TagResource",
    "secretsmanager:UntagResource"
  ],
  "Resource": "*",
  "Condition": {
    "StringNotEquals": {
      "aws:sourceVpc": "vpc-12345678"
    }
  }
```

```
},
{
  "Sid": "AllowSecretValueAccessfromONLYvpc-2b2b2b2b",
  "Effect": "Deny",
  "Principal": "*",
  "Action": [
    "secretsmanager:GetSecretValue"
  ],
  "Resource": "*",
  "Condition": {
    "StringNotEquals": {
      "aws:sourceVpc": "vpc-2b2b2b2b"
    }
  }
}
]
}
```

Contoh: Prinsip layanan

Jika kebijakan sumber daya yang dilampirkan ke rahasia Anda menyer takan prinsip AWS layanan, sebaiknya gunakan kunci kondisi SourceAccount global https://docs.aws.amazon.com/IAM/latest/UserGuide/reference_policies_condition-keys.html#condition-keys-sourcearn aws: SourceArn dan aws:. ARN dan nilai akun disertakan dalam konteks otorisasi hanya ketika permintaan datang ke Manajer Rahasia dari AWS layanan lain. Kombinasi kondisi ini menghindari skenario wakil yang mungkin membingungkan.

Jika ARN sumber daya menyertakan karakter yang tidak diizinkan dalam kebijakan sumber daya, Anda tidak dapat menggunakan ARN sumber daya tersebut dalam nilai kunci `aws:SourceArn` kondisi. Sebagai gantinya, gunakan kunci `aws:SourceAccount` kondisi. Untuk informasi selengkapnya, lihat persyaratan https://docs.aws.amazon.com/IAM/latest/UserGuide/reference_iam-quotas.html#reference_iam-quotas-names IAM.

Prinsip layanan biasanya tidak digunakan sebagai prinsipal dalam kebijakan yang melekat pada rahasia, tetapi beberapa AWS layanan memerlukannya. Untuk informasi tentang kebijakan sumber daya yang diperlukan layanan untuk melampirkan ke rahasia, lihat dokumentasi layanan.

Example Izinkan layanan untuk mengakses rahasia menggunakan prinsipal layanan

JSON

```
{
"Version":"2012-10-17",
"Statement": [
{
  "Effect": "Allow",
  "Principal": {
    "Service": [
      "s3.amazonaws.com"
    ]
  },
  "Action": "secretsmanager:GetSecretValue",
  "Resource": "*",
  "Condition": {
    "ArnLike": {
      "aws:sourceArn": "arn:aws:s3::123456789012:*"
    },
    "StringEquals": {
      "aws:sourceAccount": "123456789012"
    }
  }

}
]
}
```

Kontrol akses ke rahasia menggunakan kontrol akses berbasis atribut (ABAC)

Attribute-based Access Control (ABAC) adalah strategi otorisasi yang mendefinisikan izin berdasarkan atribut atau karakteristik pengguna, data, atau lingkungan, seperti departemen, unit bisnis, atau faktor lain yang dapat mempengaruhi hasil otorisasi. Dalam AWS, atribut ini disebut tag.

Menggunakan tag untuk mengontrol izin sangat membantu di lingkungan yang berkembang pesat dan membantu situasi di mana manajemen kebijakan menjadi rumit. Aturan ABAC dievaluasi secara dinamis saat runtime, yang berarti bahwa akses pengguna ke aplikasi dan data dan jenis operasi yang diizinkan secara otomatis berubah berdasarkan faktor kontekstual dalam kebijakan. Misalnya,

jika pengguna mengubah departemen, akses secara otomatis disesuaikan tanpa perlu memperbarui izin atau meminta peran baru. Untuk informasi lebih lanjut, lihat: Untuk apa ABAC? AWS , Tentukan izin untuk mengakses rahasia berdasarkan tag. , dan Skalakan kebutuhan otorisasi Anda untuk Secrets Manager menggunakan ABAC dengan IAM Identity Center.

Contoh: Izinkan akses identitas ke rahasia yang memiliki tag tertentu

Kebijakan berikut memungkinkan `DescribeSecret` akses pada rahasia dengan tag dengan kunci *`ServerName`* dan nilainya*`ServerABC`*. Jika Anda melampirkan kebijakan ini ke identitas, identitas tersebut memiliki izin untuk rahasia apa pun dengan tag tersebut di akun.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
  "Effect": "Allow",
  "Action": "secretsmanager:DescribeSecret",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "secretsmanager:ResourceTag/ServerName": "ServerABC"
      }
    }
  }
}
```

Contoh: Izinkan akses hanya ke identitas dengan tag yang cocok dengan tag rahasia

Kebijakan berikut memungkinkan identitas apa pun di akun `GetSecretValue` mengakses rahasia apa pun di akun di mana *`AccessProject`* tag identitas memiliki nilai yang sama dengan *`AccessProject`* tag rahasia.

JSON

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

```
  "Principal": {
    "AWS": "123456789012"
  },
  "Condition": {
    "StringEquals": {
      "aws:ResourceTag/AccessProject": "${ aws:PrincipalTag/AccessProject }"
    }
  },
  "Action": "secretsmanager:GetSecretValue",
  "Resource": "*"
  }
}
```

AWS kebijakan yang dikelola untuk AWS Secrets Manager

Kebijakan AWS terkelola adalah kebijakan mandiri yang dibuat dan dikelola oleh AWS. AWS kebijakan terkelola dirancang untuk memberikan izin untuk banyak kasus penggunaan umum sehingga Anda dapat mulai menetapkan izin kepada pengguna, grup, dan peran.

Perlu diingat bahwa kebijakan ter AWS kelola mungkin tidak memberikan izin hak istimewa terkecil untuk kasus penggunaan spesifik Anda karena kebijakan tersebut tersedia untuk digunakan semua AWS pelanggan. Kami menyarankan Anda untuk mengurangi izin lebih lanjut dengan menentukan kebijakan yang dikelola pelanggan yang khusus untuk kasus penggunaan Anda.

Anda tidak dapat mengubah izin yang ditentukan dalam kebijakan AWS terkelola. Jika AWS memperbarui izin yang ditentukan dalam kebijakan AWS terkelola, pembaruan akan memengaruhi semua identitas utama (pengguna, grup, dan peran) yang dilampirkan kebijakan tersebut. AWS kemungkinan besar akan memperbarui kebijakan ter AWS kelola ketika yang baru Layanan AWS diluncurkan atau operasi API baru tersedia untuk layanan yang ada.

Untuk informasi selengkapnya, lihat Kebijakan terkelola AWS dalam Panduan Pengguna IAM.

AWS kebijakan yang dikelola: SecretsManagerReadWrite

Kebijakan ini memberikan read/write akses ke AWS Secrets Manager, termasuk izin untuk menjelaskan sumber daya Amazon RDS, Amazon Redshift, dan Amazon DocumentDB, dan izin untuk digunakan untuk mengenkripsi dan men AWS KMS dekripsi rahasia. Kebijakan ini juga memberikan izin untuk membuat kum AWS CloudFormation pulan perubahan, mendapatkan template rotasi dari bucket Amazon S3 yang dikelola oleh AWS, daftar AWS Lambda fungsi, dan

menjelaskan VPC Amazon EC2. Izin ini diperlukan oleh konsol untuk mengatur rotasi dengan fungsi rotasi yang ada.

Untuk membuat fungsi rotasi baru, Anda juga harus memiliki izin untuk membuat tumpu AWS CloudFormation kan dan peran AWS Lambda eksekusi. Anda dapat menetapkan kebijakan ter IAMFullAccess kelola. Lihat Izin untuk rotasi.

Detail izin

Kebijakan ini mencakup izin berikut.

- `secretsmanager`- Memungkinkan kepala sekolah untuk melakukan semua tindakan Manajer Rahasia.
- `cloudformation`- Memungkinkan kepala sekolah untuk membuat tumpukan CloudFormation . Ini diperlukan agar prinsipal yang menggunakan konsol untuk mengaktifkan rotasi dapat membuat fungsi rotasi Lambda melalui CloudFormation tumpukan. Untuk informasi selengkapnya, lihat the section called “Bagaimana Secrets Manager menggunakan CloudFormation”.
- `ec2`- Memungkinkan kepala sekolah untuk menggambarkan Amazon EC2 VPC. Ini diperlukan agar prinsipal yang menggunakan konsol dapat membuat fungsi rotasi di VPC yang sama dengan database kredenSIAL yang mereka simpan secara rahasia.
- `kms`- Memungkinkan prinsipal untuk menggunakan AWS KMS kunci untuk operasi kriptografi. Ini diperlukan agar Secrets Manager dapat mengenkripsi dan mendekripsi rahasia. Untuk informasi selengkapnya, lihat the section called “Enkripsi rahasia dan dekripsi”.
- `lambda`- Memungkinkan kepala sekolah untuk mencantumkan fungsi rotasi Lambda. Ini diperlukan agar prinsipal yang menggunakan konsol dapat memilih fungsi rotasi yang ada.
- `rds`- Memungkinkan prinsipal untuk menggambarkan cluster dan instance di Amazon RDS. Ini diperlukan agar prinsipal yang menggunakan konsol dapat memilih cluster atau instans Amazon RDS.
- `redshift`- Memungkinkan kepala sekolah untuk menggambarkan cluster di Amazon Redshift. Ini diperlukan agar kepala sekolah yang menggunakan konsol dapat memilih cluster Amazon Redshift.
- `redshift-serverless`- Mengizinkan prinsipal untuk menggambarkan ruang nama di Amazon Redshift Serverless. Ini diperlukan agar prinsipal yang menggunakan konsol dapat memilih ruang nama Amazon Redshift Serverless.

- `docdb-elastic`- Memungkinkan prinsipal untuk menggambarkan cluster elastis di Amazon DocumentDB. Ini diperlukan agar prinsipal yang menggunakan konsol dapat memilih cluster elastis Amazon DocumentDB.
- `tag`— Memungkinkan kepala sekolah untuk mendapatkan semua sumber daya di akun yang ditandai.
- `serverlessrepo`- Memungkinkan kepala sekolah untuk membuat set CloudFormation perubahan. Ini diperlukan agar prinsipal yang menggunakan konsol dapat membuat fungsi rotasi Lambda. Untuk informasi selengkapnya, lihat the section called “Bagaimana Secrets Manager menggunakan CloudFormation”.
- `s3`— Memungkinkan prinsipal untuk mendapatkan objek dari bucket Amazon S3 yang dikelola oleh AWS. Ember ini berisi LambdaTemplat fungsi rotasi. Izin ini diperlukan agar prinsipal yang menggunakan konsol dapat membuat fungsi rotasi Lambda berdasarkan template di bucket. Untuk informasi selengkapnya, lihat the section called “Bagaimana Secrets Manager menggunakan CloudFormation”.

Untuk melihat kebijakan, lihat dokumen kebijakan SecretsManagerReadWrite JSON.

AWS kebijakan yang dikelola: AWSSecretsManagerClientReadOnlyAccess

Kebijakan ini menyediakan akses baca-saja ke AWS Secrets Manager rahasia untuk aplikasi klien. Ini memungkinkan prinsipal untuk mengambil nilai rahasia secara individual atau dalam batch, mencantumkan rahasia, dan menjelaskan metadata rahasia, bersama dengan AWS KMS izin yang diperlukan untuk mendekripsi rahasia yang dienkripsi dengan kunci yang dikelola pelanggan.

Detail izin

Kebijakan ini mencakup izin berikut.

- `secretsmanager`- Memungkinkan kepala sekolah untuk mengambil nilai rahasia secara individual atau dalam batch, mencantumkan rahasia, dan menjelaskan metadata rahasia.
- `kms`- Memungkinkan prinsipal untuk mendekripsi rahasia menggunakan AWS KMS kunci. Izin ini dicakup untuk kunci yang digunakan oleh Manajer Rahasia melalui kondisi khusus layanan.

Untuk melihat detail selengkapnya tentang kebijakan, termasuk versi terbaru dari dokumen kebijakan JSON, lihat AWSSecretsManagerClientReadOnlyAccess di Panduan Referensi AWS Kebijakan Terkelola.

Pembaruan Manajer Rahasia ke AWS kebijakan terkelola

Lihat detail tentang pembaruan kebijakan ter AWS kelola untuk Manajer Rahasia.

Ubah	Deskripsi	Date	Versi
SecretsManagerReadWrite – Pembaruan ke kebijakan yang ada	Kebijakan ini diperbarui untuk menambah `cloudformation:TagResource` dan mengiz `cloudformation:UntagResource` inkan. K `cloudformation:CreateAction` unci kondisi mencakup izin ini. Mereka hanya berlaku selama pembuatan atau pembaruan tumpukan melalui `ExecuteChangeSet` tindakan `CreateChangeSet` dan yang sudah diizinkan oleh kebijakan ini.	Agustus 25, 2026	v6
AWSSecretsManagerClientReadOnlyAccess – Pembaruan ke kebijakan yang ada	Kebijakan ini diperbarui untuk menambah `BatchGetSecretValue` dan `ListSecrets` mengizinkan aplikasi	02 Juni 2026	v4

Ubah	Deskripsi	Date	Versi
	klien mengambil beberapa rahasia dalam satu panggilan.		
AWSSecretsManagerClientReadOnlyAccess— Kebijakan terkelola baru	Secrets Manager membuat kebijakan terkelola baru untuk menyediakan akses baca-saja ke rahasia untuk aplikasi klien. Kebijakan ini memungkinkan pengambilan nilai rahasia dan deskripsi metadata rahasia, dengan AWS KMS izin yang diperlukan untuk mendekripsi rahasia.	5 November 2025	v1
SecretsManagerReadWrite – Pembaruan ke kebijakan yang ada	Kebijakan ini diperbarui untuk mengizinkan akses deskripsi ke Amazon Redshift Serverless sehingga pengguna konsol dapat memilih namespace Amazon Redshift Serverless saat mereka membuat rahasia Amazon Redshift.	Maret 12, 2024	v5

Ubah	Deskripsi	Date	Versi
SecretsManagerReadWrite – Pembaruan ke kebijakan yang ada	Kebijakan ini diperbarui untuk mengizinkan akses deskripsi ke cluster elastis Amazon DocumentDB sehingga pengguna konsol dapat memilih cluster elastis saat mereka membuat rahasia Amazon DocumentDB.	12 September 2023	v4
SecretsManagerReadWrite – Pembaruan ke kebijakan yang ada	Kebijakan ini diperbarui untuk mengizinkan akses deskripsi ke Amazon Redshift sehingga pengguna konsol dapat memilih cluster Amazon Redshift saat mereka membuat rahasia Amazon Redshift. Pembaruan juga menambahk an izin baru untuk memungkinkan akses baca ke bucket Amazon S3 yang dikelola oleh AWS yang menyimpan template fungsi rotasi Lambda.	24 Juni 2020	v3

Ubah	Deskripsi	Date	Versi
SecretsManagerReadWrite – Pembaruan ke kebijakan yang ada	Kebijakan ini diperbarui untuk mengizinkan akses deskripsi ke cluster Amazon RDS sehingga pengguna konsol dapat memilih cluster saat mereka membuat rahasia Amazon RDS.	3 Mei 2018	v2
SecretsManagerReadWrite – Kebijakan baru	Manajer Rahasia membuat kebijakan untuk memberikan izin yang diperlukan untuk menggunakan konsol dengan semua read/write akses ke Manajer Rahasia.	4 April 2018	v1

Tentukan siapa yang memiliki izin untuk Anda AWS Secrets Manager rahasia

Secara default, identitas IAM tidak memiliki izin untuk mengakses rahasia. Saat mengotorisasi akses ke rahasia, Manajer Rahasia mengevaluasi kebijakan berbasis sumber daya yang dilampirkan pada rahasia dan semua kebijakan berbasis identitas yang dilampirkan ke pengguna atau peran IAM yang mengirim permintaan. Untuk melakukan ini, Secrets Manager menggunakan proses yang mirip dengan yang dijelaskan dalam Menentukan apakah permintaan diizinkan atau ditolak dalam Panduan Pengguna IAM.

Ketika beberapa kebijakan diterapkan pada permintaan, Manajer Rahasia menggunakan hierarki untuk mengontrol izin:

1. Jika pernyataan dalam kebijakan apa pun dengan eksplisit `deny` cocok dengan tindakan permintaan dan sumber daya:

Yang eksplisit `deny` mengesampingkan semua yang lain dan memblokir tindakan.

2. Jika tidak ada eksplisit`deny`, tetapi pernyataan dengan eksplisit `allow` cocok dengan tindakan permintaan dan sumber daya:

   Yang ekspl `allow` isit memberikan tindakan dalam permintaan akses ke sumber daya dalam pernyataan.

   Jika identitas dan rahasia berada di dua akun yang berbeda, harus ada kebijakan sumber daya untuk rahasia dan kebijakan yang `allow` melekat pada identitas, jika tidak AWS menolak permintaan tersebut. Untuk informasi selengkapnya, lihat Cross-accountakses.

3. Jika tidak ada pernyataan dengan eksplisit `allow` yang cocok dengan tindakan permintaan dan sumber daya:

   AWS menolak permintaan secara default, yang disebut penolakan implisit.

Untuk melihat kebijakan berbasis sumber daya untuk rahasia

- Lakukan salah satu tindakan berikut:
  - Buka konsol Secrets Manager di https://console.aws.amazon.com/secretsmanager/. Di halaman detail rahasia untuk rahasia Anda, di bagian Izin sumber daya, pilih Edit izin.
  - Gunakan AWS CLI untuk memanggil `get-resource-policy` atau AWS SDK untuk menelepon `GetResourcePolicy`.

Untuk menentukan siapa yang memiliki akses melalui kebijakan berbasis identitas

- Gunakan simulator kebijakan IAM. Lihat Menguji kebijakan IAM dengan simulator kebijakan IAM

Akses AWS Secrets Manager rahasia dari akun yang berbeda

Untuk memungkinkan pengguna dalam satu akun mengakses rahasia di akun lain (akses lintas akun), Anda harus mengizinkan akses baik dalam kebijakan sumber daya maupun dalam kebijakan identitas. Ini berbeda dengan memberikan akses ke identitas di akun yang sama dengan rahasia.

Cross-account izin hanya efektif untuk operasi berikut:

- CancelRotateSecret

- DeleteResourcePolicy
- DeleteSecret
- DescribeSecret
- GetRandomPassword
- GetResourcePolicy
- GetSecretValue
- ListSecretVersionIds
- PutResourcePolicy
- PutSecretValue
- RemoveRegionsFromReplication
- ReplicateSecretToRegions
- RestoreSecret
- RotateSecret
- StopReplicationToReplica
- TagResource
- UntagResource
- UpdateSecret
- UpdateSecretVersionStage
- ValidateResourcePolicy

Anda dapat menggunakan `BlockPublicPolicy` parameter dengan PutResourcePolicytindakan untuk membantu melindungi sumber daya Anda dengan mencegah akses publik diberikan melalui kebijakan sumber daya yang secara langsung melekat pada rahasia Anda. Anda juga dapat menggunakan IAM Access Analyzer untuk memverifikasi akses lintas akun.

Anda juga harus mengizinkan identitas untuk menggunakan kunci KMS yang rahasianya dienkripsi. Ini karena Anda tidak dapat menggunakan Kunci yang dikelola AWS (`aws/secretsmanager`) untuk akses lintas akun. Sebagai gantinya, Anda harus mengenkripsi rahasia Anda dengan kunci KMS yang Anda buat, lalu lampirkan kebijakan kunci ke dalamnya. Ada biaya untuk membuat kunci KMS. Untuk mengubah kunci enkripsi untuk rahasia, lihatthe section called “Ubah rahasia”.

> **Important**
>
> Resource-based kebijakan yang memberikan `secretsmanager:PutResourcePolicy` izin memberi kepala sekolah, bahkan yang ada di akun lain, kemampuan untuk mengubah kebijakan berbasis sumber daya Anda. Izin ini memungkinkan kepala sekolah meningkatkan izin yang ada seperti mendapatkan akses administratif penuh ke rahasia. Kami menyarankan Anda menerapkan prinsip akses paling tidak istimewa ke kebijakan Anda. Untuk informasi selengkapnya, lihat Resource-based kebijakan.

Contoh kebijakan berikut mengasumsikan Anda memiliki kunci rahasia dan enkripsi di Account1, dan identitas di Account2 yang ingin Anda izinkan untuk mengakses nilai rahasia.

Langkah 1: Lampirkan kebijakan sumber daya ke rahasia di Akun1

- Kebijakan berikut memungkinkan *`ApplicationRole`* masuk *`Account2`* untuk mengakses rahasia di*`Account1`*. Untuk menggunakan kebijakan ini, lihatthe section called “Resource-based kebijakan”.

  JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:role/ApplicationRole"
            },
            "Action": "secretsmanager:GetSecretValue",
            "Resource": "*"
        }
    ]
}
```

Langkah 2: Tambahkan pernyataan ke kebijakan kunci untuk kunci KMS di Akun1

- Pernyataan kebijakan kunci berikut memungkinkan *`ApplicationRole`* *`Account2`* untuk menggunakan kunci KMS *`Account1`* untuk mendekripsi rahasia di. *`Account1`* Untuk

menggunakan pernyataan ini, tambahkan ke kebijakan kunci untuk kunci KMS Anda. Untuk informasi selengkapnya, lihat Mengubah kebijakan utama. Izin kunci KMS yang dibutuhkan prinsipal bergantung pada operasi yang dilakukannya pada rahasia. Untuk membaca nilai rahasia, kebutuhan pokokkms:Decrypt. Jika prinsipal juga membuat atau mengubah nilai rahasia (misalnya, denganPutSecretValue,UpdateSecret, atauCreateSecret), berikan jugakms:GenerateDataKey. Untuk izin kunci KMS yang diperlukan oleh setiap operasi, lihat. the section called “Izin untuk kunci KMS”

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::Account2:role/ApplicationRole"
  },
  "Action": [
    "kms:Decrypt",
    "kms:DescribeKey"
  ],
  "Resource": "*"
}
```

Langkah 3: Lampirkan kebijakan identitas ke identitas di Akun2

- Kebijakan berikut memungkinkan *ApplicationRole* masuk *Account2* untuk mengakses rahasia *Account1* dan mendekripsi nilai rahasia dengan menggunakan kunci enkripsi yang juga ada di. *Account1* Untuk menggunakan kebijakan ini, lihatthe section called “Identity-based kebijakan”. Anda dapat menemukan ARN untuk rahasia Anda di konsol Secrets Manager di halaman detail rahasia di bawah Rahasia ARN. Atau, Anda dapat menelepon describe-secret.

  JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "secretsmanager:GetSecretValue",
            "Resource": "arn:aws:secretsmanager:us-
east-1:123456789012:secret:secretName-AbCdEf"
```

```
        },
        {
            "Effect": "Allow",
            "Action": "kms:Decrypt",
            "Resource": "arn:aws:kms:us-
east-1:123456789012:key/EncryptionKey"
        }
    ]
}
```

Mengakses rahasia dari lingkungan lokal

Anda dapat menggunakan AWS Identity and Access Management Roles Anywhere untuk mendapatkan kredensi keamanan sementara di IAM untuk beban kerja seperti server, kontainer, dan aplikasi yang berjalan di luar. AWS Beban kerja Anda dapat menggunakan kebijakan IAM dan peran IAM yang sama yang Anda gunakan dengan AWS aplikasi untuk mengakses sumber daya. AWS Dengan IAM Roles Anywhere, Anda dapat menggunakan Secrets Manager untuk menyimpan dan mengelola kredensi yang dapat diakses oleh sumber daya di AWS serta perangkat lokal seperti server aplikasi. Untuk informasi selengkapnya, lihat Panduan Pengguna IAM Roles Anywhere.

Perlindungan data di AWS Secrets Manager

Model tanggung jawab AWS bersama model berlaku untuk perlindungan data di AWS Secrets Manager. Seperti yang dijelaskan dalam model AWS ini, bertanggung jawab untuk melindungi infrastruktur global yang menjalankan semua AWS Cloud. Anda harus bertanggung jawab untuk memelihara kendali terhadap konten yang di-hosting pada infrastruktur ini. Konten ini meliputi konfigurasi keamanan dan tugas-tugas pengelolaan untuk berbagai layanan Layanan AWS yang Anda gunakan. Lihat informasi yang lebih lengkap tentang privasi data dalam Pertanyaan Umum Privasi Data. Lihat informasi tentang perlindungan data di Eropa di pos blog Model Tanggung Jawab Bersama dan GDPR AWS di Blog Keamanan AWS .

Untuk tujuan perlindungan data, kami menyarankan Anda untuk melindungi Akun AWS kredensyal 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 otentikasi multi-faktor (MFA) dengan setiap akun.

- Gunakan SSL/TLS untuk berkomunikasi dengan AWS sumber daya. Secrets Manager mendukung TLS 1.2 dan 1.3 di semua Wilayah. Secrets Manager juga mendukung opsi pertukaran kunci pasca-kuantum hibrida untuk protokol enkripsi jaringan TLS (PQTLS).
- Tanda tangani permintaan terprogram Anda ke Secrets Manager dengan 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 kredensyal keamanan sementara untuk menandatangani permintaan.
- Siapkan API dan pencatatan aktivitas pengguna dengan AWS CloudTrail. Lihat the section called “Masuk dengan AWS CloudTrail ”.
- Jika Anda memerlukan modul kriptografi tervalidasi FIPS 140-2 saat mengakses AWS melalui antarmuka baris perintah atau API, gunakan titik akhir FIPS. Lihat the section called “Titik akhir Manajer Rahasia”.
- Jika Anda menggunakan AWS CLI untuk mengakses Secrets Manager,the section called “Mengurangi risiko menggunakan AWS CLI untuk menyimpan rahasia Anda AWS Secrets Manager”.

Enkripsi saat diam

Secrets Manager menggunakan enkripsi via AWS Key Management Service (AWS KMS) untuk melindungi kerahasiaan data saat istirahat. AWS KMS menyediakan penyimpanan kunci dan layanan enkripsi yang digunakan oleh banyak AWS layanan. Setiap rahasia di Secrets Manager dienkripsi dengan kunci data yang unik. Setiap kunci data dilindungi oleh kunci KMS. Anda dapat memilih untuk menggunakan enkripsi default dengan Secrets Manager Kunci yang dikelola AWS untuk akun, atau Anda dapat membuat kunci terkelola pelanggan Anda sendiri AWS KMS. Menggunakan kunci yang dikelola pelanggan memberi Anda kontrol otorisasi yang lebih terperinci atas aktivitas utama KMS Anda. Untuk informasi selengkapnya, lihat the section called “Enkripsi rahasia dan dekripsi”.

Enkripsi saat bergerak

Secrets Manager menyediakan endpoint yang aman dan pribadi untuk mengenkripsi data dalam perjalanan. Endpoint yang aman dan pribadi memungkinkan AWS untuk melindungi integritas permintaan API ke Secrets Manager. AWS memerlukan panggilan API ditandatangani oleh pemanggil menggunakan X.509 sertifikat and/or Secrets Manager Secret Access Key. Persyaratan ini dinyatakan dalam Proses Penandatanganan Versi Tanda Tangan 4 (Sigv4).

Jika Anda menggunakan AWS Command Line Interface (AWS CLI) atau salah satu AWS SDK untuk melakukan panggilan AWS, Anda mengonfigurasi kunci akses yang akan digunakan. Kemudian alat-

alat tersebut secara otomatis menggunakan tombol akses untuk menandatangani permintaan untuk Anda. Lihat the section called “Mengurangi risiko menggunakan AWS CLI untuk menyimpan rahasia Anda AWS Secrets Manager”.

Inter-network privasi lalu lintas

AWS menawarkan opsi untuk menjaga privasi saat merutekan lalu lintas melalui rute jaringan yang dikenal dan pribadi.

Lalu lintas antara layanan dan aplikasi serta klien on-premise

Anda memiliki dua opsi konektivitas antara jaringan pribadi Anda dan AWS Secrets Manager:

- Koneksi AWS Site-to-Site VPN. Untuk informasi selengkapnya, lihat Apa itu AWS Site-to-Site VPN?
- Koneksi AWS Direct Connect. Untuk informasi selengkapnya, lihat Apa itu AWS Direct Connect?

Lalu lintas antar AWS sumber daya di Wilayah yang sama

Jika Anda ingin mengamankan lalu lintas antara Secrets Manager dan klien API AWS, siapkan AWS PrivateLinkuntuk mengakses titik akhir Secrets Manager API secara pribadi.

Pengelolaan kunci enkripsi

Ketika Secrets Manager perlu mengenkripsi versi baru dari data rahasia yang dilindungi, Secrets Manager mengirimkan permintaan AWS KMS untuk menghasilkan kunci data baru dari kunci KMS. Secrets Manager menggunakan kunci data ini untuk enkripsi amplop. Secrets Manager menyimpan kunci data terenkripsi dengan rahasia terenkripsi. Ketika rahasia perlu didekripsi, Secrets Manager meminta AWS KMS untuk mendekripsi kunci data. Secrets Manager kemudian menggunakan kunci data yang didekripsi untuk mendekripsi rahasia terenkripsi. Secrets Manager tidak pernah menyimpan kunci data dalam bentuk yang tidak terenkripsi dan menghapus kunci dari memori sesegera mungkin. Lihat informasi yang lebih lengkap di the section called “Enkripsi rahasia dan dekripsi”.

Enkripsi rahasia dan dekripsi di AWS Secrets Manager

Secrets Manager menggunakan enkripsi amplop dengan AWS KMS kunci dan kunci data untuk melindungi setiap nilai rahasia. Setiap kali nilai rahasia dalam rahasia berubah, Secrets Manager meminta kunci data baru AWS KMS untuk melindunginya. Kunci data dienkripsi di bawah kunci KMS

dan disimpan dalam metadata rahasia. Untuk mendekripsi rahasia, Secrets Manager terlebih dahulu mendekripsi kunci data terenkripsi menggunakan kunci KMS. AWS KMS

Secrets Manager tidak menggunakan kunci KMS untuk mengenkripsi nilai rahasia secara langsung. Sebaliknya, ia menggunakan kunci KMS untuk menghasilkan dan mengenkripsi kunci data simetris Advanced Encryption Standard (AES) 256-bit, dan menggunakan kunci data untuk mengenkripsi nilai rahasia. Secrets Manager menggunakan kunci data plaintext untuk mengenkripsi nilai rahasia di luar AWS KMS, dan kemudian menghapusnya dari memori. Ini menyimpan salinan terenkripsi dari kunci data dalam metadata dari rahasia.

Topik

- Memilih AWS KMS kunci
- Apa yang dienkripsi?
- Proses enkripsi dan dekripsi
- Izin untuk kunci KMS
- Bagaimana Secrets Manager menggunakan kunci KMS Anda
- Kebijakan utama dari Kunci yang dikelola AWS (aws/secretsmanager)
- Konteks enkripsi Secrets Manager
- Memantau interaksi Secrets Manager dengan AWS KMS

Memilih AWS KMS kunci

Saat Anda membuat rahasia, Anda dapat memilih kunci yang dikelola pelanggan enkripsi simetris di Akun AWS dan Wilayah, atau Anda dapat menggunakan Kunci yang dikelola AWS for Secrets Manager (`aws/secretsmanager`). Jika Anda memilih Kunci yang dikelola AWS `aws/secretsmanager` dan itu belum ada, Secrets Manager membuatnya dan mengaitkannya dengan rahasia. Anda dapat menggunakan kunci KMS yang sama atau kunci KMS yang berbeda untuk setiap rahasia di akun Anda. Anda mungkin ingin menggunakan kunci KMS yang berbeda untuk mengatur izin khusus pada kunci untuk sekelompok rahasia, atau jika Anda ingin mengaudit operasi tertentu untuk kunci tersebut. Secrets Manager hanya mendukung kunci KMS enkripsi simetris. Jika Anda menggunakan kunci KMS di toko kunci eksternal, operasi kriptografi pada kunci KMS mungkin memakan waktu lebih lama dan kurang dapat diandalkan dan tahan lama karena permintaan harus melakukan perjalanan di luar. AWS

Untuk informasi tentang mengubah kunci enkripsi untuk rahasia, lihatthe section called “Mengubah kunci enkripsi untuk rahasia”.

Saat Anda mengubah kunci enkripsi, Secrets Manager mengenkripsi ulang `AWSCURRENTAWSPENDING`, dan `AWSPREVIOUS` versi dengan kunci baru. Untuk menghindari mengunci Anda dari rahasia, Secrets Manager menyimpan semua versi yang ada dienkripsi dengan kunci sebelumnya. Itu berarti Anda dapat mendekripsi`AWSCURRENT`,`AWSPENDING`, dan `AWSPREVIOUS` versi dengan kunci sebelumnya atau kunci baru. Jika Anda tidak memiliki `kms:Decrypt` izin untuk kunci sebelumnya, ketika Anda mengubah kunci enkripsi, Secrets Manager tidak dapat mendekripsi versi rahasia untuk mengenkripsi ulang mereka. Dalam hal ini, versi yang ada tidak dienkripsi ulang.

Untuk membuatnya sehingga hanya `AWSCURRENT` dapat didekripsi oleh kunci enkripsi baru, buat versi baru rahasia dengan kunci baru. Kemudian untuk dapat mendekripsi versi `AWSCURRENT` rahasia, Anda harus memiliki izin untuk kunci baru.

Anda dapat menolak izin untuk Kunci yang dikelola AWS `aws/secretsmanager` dan memerlukan rahasia dienkripsi dengan kunci yang dikelola pelanggan. Untuk informasi selengkapnya, lihat the section called “Contoh: Tolak AWS KMS kunci tertentu untuk mengenkripsi rahasia”.

Untuk menemukan kunci KMS yang terkait dengan rahasia, lihat rahasia di konsol atau panggil ListSecretsatau DescribeSecret. Ketika rahasia dikaitkan dengan Kunci yang dikelola AWS for Secrets Manager (`aws/secretsmanager`), operasi ini tidak mengembalikan pengenal kunci KMS.

Apa yang dienkripsi?

Secrets Manager mengenkripsi nilai rahasia, tetapi tidak mengenkripsi yang berikut:

- Nama dan deskripsi rahasia
- Pengaturan rotasi
- ARN dari kunci KMS yang terkait dengan rahasia
- Setiap AWS tag terlampir

Proses enkripsi dan dekripsi

Untuk mengenkripsi nilai rahasia dalam rahasia, Secrets Manager menggunakan proses berikut.

1. Secrets Manager memanggil AWS KMS GenerateDataKeyoperasi dengan ID kunci KMS untuk rahasia dan permintaan untuk kunci simetris AES 256-bit. AWS KMS mengembalikan kunci data plaintext dan salinan kunci data yang dienkripsi di bawah kunci KMS.

2. Secrets Manager menggunakan kunci data plaintext dan algoritma Advanced Encryption Standard (AES) untuk mengenkripsi nilai rahasia di luar. AWS KMS Ini akan menghapus kunci plaintext dari memori sesegera mungkin setelah menggunakannya.
3. Secrets Manager menyimpan kunci data terenkripsi dalam metadata rahasia sehingga ini tersedia untuk mendekripsi nilai rahasia. Namun, tidak ada API Secrets Manager yang mengembalikan rahasia terenkripsi atau kunci data terenkripsi.

Untuk mendekripsi nilai rahasia terenkripsi:

1. Secrets Manager memanggil operasi AWS KMS Dekripsi dan meneruskan kunci data terenkripsi.
2. AWS KMS menggunakan kunci KMS untuk rahasia untuk mendekripsi kunci data. Ini mengembalikan kunci data plaintext.
3. Secrets Manager menggunakan kunci data plaintext untuk mendekripsi nilai rahasia. Kemudian, ini menghapus kunci data dari memori sesegera mungkin.

Izin untuk kunci KMS

Ketika Secrets Manager menggunakan kunci KMS dalam operasi kriptografi, ia bertindak atas nama pengguna yang mengakses atau memperbarui nilai rahasia. Anda dapat memberikan izin dalam kebijakan IAM atau kebijakan utama. Operasi Secrets Manager berikut memerlukan AWS KMS izin.

- CreateSecret
- GetSecretValue
- PutSecretValue
- UpdateSecret
- ReplicateSecretToRegions

Untuk mengizinkan kunci KMS hanya digunakan untuk permintaan yang berasal dari Secrets Manager, dalam kebijakan izin, Anda dapat menggunakan kunci ViaService kondisi kms: dengan nilainya. `secretsmanager.<Region>.amazonaws.com`

Anda juga dapat menggunakan kunci atau nilai dalam konteks enkripsi sebagai syarat untuk menggunakan kunci KMS untuk operasi kriptografi. Misalnya, Anda dapat menggunakan operator ketentuan string di IAM atau dokumen kebijakan kunci, atau menggunakan batasan hibah dalam

hibah. Perbanyakan hibah kunci KMS dapat memakan waktu hingga lima menit. Untuk informasi selengkapnya, lihat CreateGrant.

Bagaimana Secrets Manager menggunakan kunci KMS Anda

Secrets Manager memanggil AWS KMS operasi berikut dengan kunci KMS Anda.

GenerateDataKey

Secrets Manager memanggil AWS KMS GenerateDataKeyoperasi sebagai tanggapan atas operasi Secrets Manager berikut.

- CreateSecretJika rahasia baru menyertakan nilai rahasia, Secrets Manager meminta kunci data baru untuk mengenkripsinya.
- PutSecretValue— Secrets Manager meminta kunci data baru untuk mengenkripsi nilai rahasia yang ditentukan.
- ReplicateSecretToRegions— Untuk mengenkripsi rahasia yang direplikasi, Secrets Manager meminta kunci data untuk kunci KMS di Region replika.
- UpdateSecret— Jika Anda mengubah nilai rahasia atau kunci KMS, Secrets Manager meminta kunci data baru untuk mengenkripsi nilai rahasia baru.

RotateSecretOperasi tidak memanggil`GenerateDataKey`, karena tidak mengubah nilai rahasia. Namun, jika `RotateSecret` memanggil fungsi rotasi Lambda yang mengubah nilai rahasia, panggilannya ke `PutSecretValue` operasi memicu `GenerateDataKey` permintaan.

Dekripsi

Secrets Manager memanggil operasi Dekripsi sebagai respons untuk operasi Secrets Manager berikut.

- GetSecretValuedan BatchGetSecretValue— Secrets Manager mendekripsi nilai rahasia sebelum mengembalikannya ke penelepon. Untuk mendekripsi nilai rahasia terenkripsi, Secrets Manager memanggil operasi Dekripsi untuk AWS KMS mendekripsi kunci data terenkripsi dalam rahasia. Kemudian, ini menggunakan kunci data plaintext untuk mendekripsi nilai rahasia terenkripsi. Untuk perintah batch, Secrets Manager dapat menggunakan kembali kunci yang didekripsi, sehingga tidak semua panggilan menghasilkan permintaan. `Decrypt`
- PutSecretValuedan UpdateSecret— Sebagian besar `PutSecretValue` dan `UpdateSecret` permintaan tidak memicu `Decrypt` operasi. Namun, ketika permintaan `PutSecretValue` atau `UpdateSecret` berusaha untuk mengubah nilai rahasia dalam versi rahasia yang ada, Secrets Manager mendekripsi nilai rahasia yang ada dan membandingkannya dengan nilai

rahasia dalam permintaan untuk mengonfirmasi bahwa mereka adalah sama. Tindakan ini memastikan bahwa operasi Secrets Manager adalah idempoten. Untuk mendekripsi nilai rahasia terenkripsi, Secrets Manager memanggil operasi Dekripsi untuk AWS KMS mendekripsi kunci data terenkripsi dalam rahasia. Kemudian, ini menggunakan kunci data plaintext untuk mendekripsi nilai rahasia terenkripsi.

- ReplicateSecretToRegions— Secrets Manager pertama kali mendekripsi nilai rahasia di Wilayah utama sebelum mengenkripsi ulang nilai rahasia dengan kunci KMS di Region replika.

Enkripsi

Secrets Manager memanggil operasi Enkripsi sebagai respons terhadap operasi Secrets Manager berikut:

- UpdateSecret— Jika Anda mengubah kunci KMS, Secrets Manager mengenkripsi ulang kunci data yang melindungi`AWSCURRENT`,`AWSPREVIOUS`, dan versi `AWSPENDING` rahasia dengan kunci baru.

DescribeKey

Secrets Manager memanggil DescribeKeyoperasi untuk menentukan apakah akan mencantumkan kunci KMS saat Anda membuat atau mengedit rahasia di konsol Secrets Manager.

Memvalidasi akses ke kunci KMS

Ketika Anda membuat atau mengubah kunci KMS yang terkait dengan rahasia, Secrets Manager memanggil `GenerateDataKey` dan `Decrypt` operasi dengan kunci KMS yang ditentukan. Panggilan ini mengonfirmasi bahwa penelepon memiliki izin untuk menggunakan kunci KMS untuk operasi ini. Secrets Manager membuang hasil operasi tersebut; itu tidak menggunakannya dalam operasi kriptografi.

Anda dapat mengidentifikasi panggilan validasi ini karena nilai dari kunci `SecretVersionId` konteks enkripsi dalam permintaan ini adalah `RequestToValidateKeyAccess`.

> **Note**
>
> Di masa lalu, panggilan validasi Secrets Manager tidak termasuk konteks enkripsi. Anda mungkin menemukan panggilan tanpa konteks enkripsi di AWS CloudTrail log lama.

Kebijakan utama dari Kunci yang dikelola AWS (`aws/secretsmanager`)

Kebijakan kunci Kunci yang dikelola AWS untuk Secrets Manager (`aws/secretsmanager`) memberi pengguna izin untuk menggunakan kunci KMS untuk operasi tertentu hanya jika Secrets Manager membuat permintaan atas nama pengguna. Kebijakan kunci tidak mengizinkan pengguna untuk menggunakan kunci KMS secara langsung.

Kebijakan utama ini, seperti kebijakan semua Kunci yang dikelola AWS, ditetapkan oleh layanan. Anda tidak dapat mengubah kebijakan kunci, tetapi Anda dapat melihatnya kapan saja. Untuk detailnya, lihat Melihat kebijakan utama.

Pernyataan kebijakan dalam kebijakan kunci memiliki efek sebagai berikut:

- Izinkan pengguna di akun untuk menggunakan kunci KMS untuk operasi kriptografi hanya ketika permintaan berasal dari Secrets Manager atas nama mereka. Kunci kondisi `kms:ViaService` memberlakukan pembatasan ini.
- Memungkinkan AWS akun untuk membuat kebijakan IAM yang memungkinkan pengguna untuk melihat properti kunci KMS dan mencabut hibah.
- Meskipun Secrets Manager tidak menggunakan hibah untuk mendapatkan akses ke kunci KMS, kebijakan ini juga memungkinkan Secrets Manager untuk membuat hibah untuk kunci KMS atas nama pengguna dan memungkinkan akun untuk mencabut hibah yang memungkinkan Secrets Manager untuk menggunakan kunci KMS. Ini adalah elemen standar dokumen kebijakan untuk sebuah Kunci yang dikelola AWS.

Berikut ini adalah kebijakan utama untuk Kunci yang dikelola AWS contoh Secrets Manager.

JSON

```
{
  "Id": "auto-secretsmanager-2",
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "Allow access through AWS Secrets Manager for all principals in the
 account that are authorized to use AWS Secrets Manager",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "*"
```

```
        ]
      },
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:CreateGrant",
        "kms:DescribeKey"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "kms:CallerAccount": "111122223333",
          "kms:ViaService": "secretsmanager.us-west-2.amazonaws.com"
        }
      }
    },
    {
      "Sid": "Allow access through AWS Secrets Manager for all principals in the
account that are authorized to use AWS Secrets Manager",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "*"
        ]
      },
      "Action": "kms:GenerateDataKey*",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "kms:CallerAccount": "111122223333"
        },
        "StringLike": {
          "kms:ViaService": "secretsmanager.us-west-2.amazonaws.com"
        }
      }
    },
    {
      "Sid": "Allow direct access to key metadata to the account",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::111122223333:root"
        ]
```

```
      },
      "Action": [
        "kms:Describe*",
        "kms:Get*",
        "kms:List*",
        "kms:RevokeGrant"
      ],
      "Resource": "*"
    }
  ]
}
```

Konteks enkripsi Secrets Manager

Konteks enkripsi adalah sekumpulan pasangan kunci-nilai yang berisi data non-rahasia arbitrer. Ketika Anda menyertakan konteks enkripsi dalam permintaan untuk mengenkripsi data, secara AWS KMS kriptografis mengikat konteks enkripsi ke data terenkripsi. Untuk mendekripsi data, Anda harus meneruskan konteks enkripsi yang sama.

Dalam permintaannya GenerateDataKeydan Dekripsi ke AWS KMS, Secrets Manager menggunakan konteks enkripsi dengan dua pasangan nama-nilai yang mengidentifikasi rahasia dan versinya, seperti yang ditunjukkan pada contoh berikut. Nama-nama tidak bervariasi, tetapi nilai-nilai konteks enkripsi gabungan akan berbeda untuk setiap nilai rahasia.

```
"encryptionContext": {
    "SecretARN": "arn:aws:secretsmanager:us-east-2:111122223333:secret:test-secret-
a1b2c3",
    "SecretVersionId": "EXAMPLE1-90ab-cdef-fedc-ba987SECRET1"
}
```

Anda dapat menggunakan konteks enkripsi untuk mengidentifikasi operasi kriptografi ini dalam catatan audit dan log, seperti AWS CloudTraildan Amazon CloudWatch Logs, dan sebagai syarat untuk otorisasi dalam kebijakan dan hibah.

Enkripsi konteks Secrets Manager terdiri dari dua pasangan nama-nilai.

- SecretARN — Pasangan nama-nilai pertama mengidentifikasi rahasia. Kuncinya adalah `SecretARN`. Nilai tersebut adalah Amazon Resource Name (ARN) dari rahasia.

  ```
  "SecretARN": "ARN of an Secrets Manager secret"
  ```

Sebagai contoh, jika ARN dari rahasia adalah `arn:aws:secretsmanager:us-east-2:111122223333:secret:test-secret-a1b2c3`, maka konteks enkripsi akan mencakup pasangan berikut.

```
"SecretARN": "arn:aws:secretsmanager:us-east-2:111122223333:secret:test-secret-a1b2c3"
```

- SecretVersionId— Pasangan nama-nilai kedua mengidentifikasi versi rahasia. Kuncinya adalah `SecretVersionId`. Nilai adalah ID versi.

  ```
  "SecretVersionId": "<version-id>"
  ```

  Sebagai contoh, jika ID versi dari rahasia adalah `EXAMPLE1-90ab-cdef-fedc-ba987SECRET1`, maka konteks enkripsi akan mencakup pasangan berikut.

  ```
  "SecretVersionId": "EXAMPLE1-90ab-cdef-fedc-ba987SECRET1"
  ```

Saat Anda membuat atau mengubah kunci KMS untuk rahasia, Secrets Manager mengirim GenerateDataKeydan Mendekripsi permintaan AWS KMS untuk memvalidasi bahwa pemanggil memiliki izin untuk menggunakan kunci KMS untuk operasi ini. Ini membuang tanggapan; tidak menggunakannya pada nilai rahasia.

Dalam permintaan validasi ini, nilai dari `SecretARN` adalah ARN sebenarnya dari rahasia, tetapi nilai `SecretVersionId` adalah `RequestToValidateKeyAccess`, seperti yang ditunjukkan dalam konteks enkripsi contoh berikut. Nilai khusus ini membantu Anda untuk mengidentifikasi permintaan validasi di log dan jejak audit.

```
"encryptionContext": {
    "SecretARN": "arn:aws:secretsmanager:us-east-2:111122223333:secret:test-secret-a1b2c3",
    "SecretVersionId": "RequestToValidateKeyAccess"
}
```

> **Note**
>
> Di masa lalu, permintaan validasi Secrets Manager tidak termasuk konteks enkripsi. Anda mungkin menemukan panggilan tanpa konteks enkripsi di AWS CloudTrail log lama.

Memantau interaksi Secrets Manager dengan AWS KMS

Anda dapat menggunakan AWS CloudTrail dan Amazon CloudWatch Logs untuk melacak permintaan yang dikirimkan Secrets Manager atas nama Anda. AWS KMS Untuk informasi tentang pemantauan penggunaan rahasia, lihatMemantau rahasia.

GenerateDataKey

Saat Anda membuat atau mengubah nilai rahasia dalam rahasia, Secrets Manager mengirimkan GenerateDataKeypermintaan AWS KMS yang menentukan kunci KMS untuk rahasia tersebut.

Peristiwa yang mencatat operasi `GenerateDataKey` serupa dengan peristiwa contoh berikut. Permintaan dipanggil oleh `secretsmanager.amazonaws.com`. Parameter termasuk Nama Sumber Daya Amazon (ARN) dari kunci KMS untuk rahasia, penentu kunci yang memerlukan kunci 256-bit, dan konteks enkripsi yang mengidentifikasi rahasia dan versi.


```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AROAIGDTESTANDEXAMPLE:user01",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/user01",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2018-05-31T23:23:41Z"
            }
        },
        "invokedBy": "secretsmanager.amazonaws.com"
    },
    "eventTime": "2018-05-31T23:23:41Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateDataKey",
    "awsRegion": "us-east-2",
    "sourceIPAddress": "secretsmanager.amazonaws.com",
    "userAgent": "secretsmanager.amazonaws.com",
    "requestParameters": {
        "keyId": "arn:aws:kms:us-
east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "keySpec": "AES_256",
```

```
        "encryptionContext": {
            "SecretARN": "arn:aws:secretsmanager:us-east-2:111122223333:secret:test-
secret-a1b2c3",
            "SecretVersionId": "EXAMPLE1-90ab-cdef-fedc-ba987SECRET1"
        }
    },
    "responseElements": null,
    "requestID": "a7d4dd6f-6529-11e8-9881-67744a270888",
    "eventID": "af7476b6-62d7-42c2-bc02-5ce86c21ed36",
    "readOnly": true,
    "resources": [
        {
            "ARN": "arn:aws:kms:us-
east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
            "accountId": "111122223333",
            "type": "AWS::KMS::Key"
        }
    ],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
}
```

Dekripsi

Ketika Anda mendapatkan atau mengubah nilai rahasia rahasia, Secrets Manager mengirimkan permintaan Dekripsi AWS KMS untuk mendekripsi kunci data terenkripsi. Untuk perintah batch, Secrets Manager dapat menggunakan kembali kunci yang didekripsi, sehingga tidak semua panggilan menghasilkan permintaan. `Decrypt`

Peristiwa yang mencatat operasi `Decrypt` serupa dengan peristiwa contoh berikut. Pengguna adalah kepala sekolah di AWS akun Anda yang mengakses tabel. Parameter termasuk kunci tabel terenkripsi (sebagai gumpalan ciphertext) dan konteks enkripsi yang mengidentifikasi tabel dan akun. AWS AWS KMS memperoleh ID kunci KMS dari ciphertext.

```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AROAIGDTESTANDEXAMPLE:user01",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/user01",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
```

```
        "sessionContext": {
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2018-05-31T23:36:09Z"
            }
        },
        "invokedBy": "secretsmanager.amazonaws.com"
    },
    "eventTime": "2018-05-31T23:36:09Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "Decrypt",
    "awsRegion": "us-east-2",
    "sourceIPAddress": "secretsmanager.amazonaws.com",
    "userAgent": "secretsmanager.amazonaws.com",
    "requestParameters": {
        "encryptionContext": {
            "SecretARN": "arn:aws:secretsmanager:us-east-2:111122223333:secret:test-
secret-a1b2c3",
            "SecretVersionId": "EXAMPLE1-90ab-cdef-fedc-ba987SECRET1"
        }
    },
    "responseElements": null,
    "requestID": "658c6a08-652b-11e8-a6d4-ffee2046048a",
    "eventID": "f333ec5c-7fc1-46b1-b985-cbda13719611",
    "readOnly": true,
    "resources": [
        {
            "ARN": "arn:aws:kms:us-
east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
            "accountId": "111122223333",
            "type": "AWS::KMS::Key"
        }
    ],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
}
```


Enkripsi

Ketika Anda mengubah kunci KMS yang terkait dengan rahasia, Secrets Manager mengirimkan permintaan Enkripsi AWS KMS untuk mengenkripsi ulang`AWSCURRENT`,`AWSPREVIOUS`, dan versi `AWSPENDING` rahasia dengan kunci baru. Saat Anda mereplikasi rahasia ke Wilayah lain, Secrets Manager juga mengirimkan permintaan Enkripsi ke. AWS KMS

Peristiwa yang mencatat operasi `Encrypt` serupa dengan peristiwa contoh berikut. Pengguna adalah kepala sekolah di AWS akun Anda yang mengakses tabel.


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AROAIGDTESTANDEXAMPLE:user01",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/user01",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "attributes": {
                "creationDate": "2023-06-09T18:11:34Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "secretsmanager.amazonaws.com"
    },
    "eventTime": "2023-06-09T18:11:34Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "Encrypt",
    "awsRegion": "us-east-2",
    "sourceIPAddress": "secretsmanager.amazonaws.com",
    "userAgent": "secretsmanager.amazonaws.com",
    "requestParameters": {
        "keyId": "arn:aws:kms:us-east-2:111122223333:key/EXAMPLE1-f1c8-4dce-8777-
aa071ddefdcc",
        "encryptionAlgorithm": "SYMMETRIC_DEFAULT",
        "encryptionContext": {
            "SecretARN": "arn:aws:secretsmanager:us-
east-2:111122223333:secret:ChangeKeyTest-5yKnKS",
            "SecretVersionId": "EXAMPLE1-5c55-4d7c-9277-1b79a5e8bc50"
        }
    },
    "responseElements": null,
    "requestID": "129bd54c-1975-4c00-9b03-f79f90e61d60",
    "eventID": "f7d9ff39-15ab-47d8-b94c-56586de4ab68",
    "readOnly": true,
    "resources": [
        {
            "accountId": "AWS Internal",
            "type": "AWS::KMS::Key",
```

```
            "ARN": "arn:aws:kms:us-west-2:111122223333:key/EXAMPLE1-f1c8-4dce-8777-
aa071ddefdcc"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

Keamanan infrastruktur di AWS Secrets Manager

Sebagai layanan terkelola, AWS Secrets Manager 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.

Akses ke Secrets Manager melalui jaringan adalah melalui API yang AWS dipublikasikan menggunakan TLS. Secrets Manager API dapat dipanggil dari lokasi jaringan mana pun. Namun, Secrets Manager mendukung kebijakan akses berbasis sumber daya, yang dapat mencakup pembatasan berdasarkan alamat IP sumber. Anda juga dapat menggunakan kebijakan sumber daya Secrets Manager untuk mengontrol akses ke rahasia dari titik akhir virtual private cloud (VPC) tertentu, atau VPC tertentu. Secara efektif, ini mengisolasi akses jaringan ke rahasia tertentu hanya dari VPC spesifik dalam AWS jaringan. Lihat informasi yang lebih lengkap di the section called “Titik akhir VPC (AWS PrivateLink)”.

Menggunakan titik akhir AWS Secrets Manager VPC

Kami menyarankan Anda menjalankan infrastruktur sebanyak mungkin di jaringan pribadi yang tidak dapat diakses dari internet publik. Anda dapat membuat koneksi pribadi antara VPC dan Secrets Manager dengan membuat antarmuka VPC endpoint. Endpoint antarmuka didukung oleh AWS PrivateLink, teknologi yang memungkinkan Anda mengakses Secrets Manager API secara pribadi tanpa gateway internet, perangkat NAT, koneksi VPN, atau koneksi. Direct Connect Instans di VPC Anda tidak memerlukan alamat IP publik untuk berkomunikasi dengan Secrets Manager API. Lalu lintas antara VPC dan Secrets Manager Anda tidak meninggalkan jaringan. AWS Untuk informasi selengkapnya, lihat Antarmuka VPC endpoint (AWS PrivateLink) dalam Panduan Pengguna Amazon VPC.

Ketika Secrets Manager memutar rahasia dengan menggunakan fungsi rotasi Lambda, misalnya rahasia yang berisi kredensi database, fungsi Lambda membuat permintaan ke database dan Secrets Manager. Saat Anda mengaktifkan rotasi otomatis menggunakan konsol, Secrets Manager membuat fungsi Lambda di VPC yang sama dengan database Anda. Kami menyarankan Anda membuat titik akhir Secrets Manager di VPC yang sama sehingga permintaan dari fungsi rotasi Lambda ke Secrets Manager tidak meninggalkan jaringan Amazon.

Jika Anda mengaktifkan DNS pribadi untuk titik akhir, Anda dapat membuat permintaan API ke Secrets Manager menggunakan nama DNS default untuk Wilayah, misalnya,. `secretsmanager.us-east-1.amazonaws.com` Untuk informasi selengkapnya, lihat Mengakses layanan melalui titik akhir antarmuka dalam Panduan Pengguna Amazon VPC.

Anda dapat memastikan bahwa permintaan ke Secrets Manager berasal dari akses VPC dengan menyertakan kondisi dalam kebijakan izin Anda. Untuk informasi selengkapnya, lihat the section called “Contoh: Izin dan VPC”.

Anda dapat menggunakan AWS CloudTrail log untuk mengaudit penggunaan rahasia Anda melalui titik akhir VPC.

Untuk membuat titik akhir VPC untuk Secrets Manager

1. Lihat Membuat titik akhir antarmuka di Panduan Pengguna Amazon VPC. Gunakan salah satu nama layanan berikut:
   - `com.amazonaws.region.secretsmanager`
   - `com.amazonaws.region.secretsmanager-fips`
2. Untuk mengontrol akses ke titik akhir, lihat Mengontrol akses ke titik akhir VPC menggunakan kebijakan titik akhir.
3. Untuk menggunakan IPv6 dan pengalamatan dual-stack, lihat. Akses IPv4 dan IPv6

Buat kebijakan titik akhir untuk titik akhir antarmuka Anda

Kebijakan endpoint adalah sumber daya IAM yang dapat Anda lampirkan ke titik akhir antarmuka. Kebijakan endpoint default memungkinkan akses penuh ke Secrets Manager melalui titik akhir antarmuka. Untuk mengontrol akses yang diizinkan ke Secrets Manager dari VPC Anda, lampirkan kebijakan endpoint khusus ke titik akhir antarmuka.

kebijakan titik akhir mencantumkan informasi berikut:

- Prinsipal yang dapat melakukan tindakan (Akun AWS, pengguna IAM, dan peran IAM).
- Tindakan yang dapat dilakukan.
- Sumber daya untuk melakukan tindakan.

Untuk informasi selengkapnya, lihat Mengontrol akses ke layanan menggunakan kebijakan titik akhir di Panduan AWS PrivateLink .

Contoh: Kebijakan titik akhir VPC untuk tindakan Secrets Manager

Berikut ini adalah contoh kebijakan endpoint kustom. Saat Anda melampirkan kebijakan ini ke titik akhir antarmuka Anda, kebijakan ini akan memberikan akses ke tindakan Secrets Manager yang terdaftar pada rahasia yang ditentukan.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "Allow all users to use GetSecretValue and DescribeSecret on the
 specified secret.",
      "Effect": "Allow",
      "Principal": "*",
      "Action": [
        "secretsmanager:GetSecretValue",
        "secretsmanager:DescribeSecret"
      ],
      "Resource": "arn:aws:secretsmanager:us-
east-1:111122223333:secret:secretName-AbCdEf"
    }
  ]
}
```

Subnet bersama

Anda tidak dapat membuat, mendeskripsikan, memodifikasi, atau menghapus titik akhir VPC di subnet yang dibagikan dengan Anda. Namun, Anda dapat menggunakan titik akhir VPC di subnet yang dibagikan dengan Anda. Untuk informasi tentang berbagi VPC, lihat Membagikan VPC Anda dengan akun lain di Panduan Pengguna Amazon Virtual Private Cloud.

Kontrol akses API dengan kebijakan IAM

Jika Anda menggunakan kebijakan IAM untuk mengontrol akses Layanan AWS berdasarkan alamat IP, Anda mungkin perlu memperbarui kebijakan Anda untuk menyertakan rentang alamat IPv6. Panduan ini menjelaskan perbedaan antara IPv4 dan IPv6 dan menjelaskan cara memperbarui kebijakan IAM Anda untuk mendukung kedua protokol. Menerapkan perubahan ini membantu Anda menjaga akses aman ke AWS sumber daya Anda sambil mendukung IPv6.

Apa itu IPv6?

IPv6 adalah standar IP generasi berikutnya yang dimaksudkan untuk akhirnya menggantikan IPv4. Versi sebelumnya, IPv4, menggunakan skema pengalamatan 32-bit untuk mendukung 4,3 miliar perangkat. IPv6 malah menggunakan pengalamatan 128-bit untuk mendukung sekitar 340 triliun triliun triliun (atau 2 hingga daya 128) perangkat.

Untuk informasi selengkapnya, lihat halaman web VPC IPv6.

Ini adalah contoh alamat IPv6:

```
2001:cdba:0000:0000:0000:0000:3257:9652 # This is a full, unabbreviated IPv6 address.
2001:cdba:0:0:0:0:3257:9652             # The same address with leading zeros in each
 group omitted
2001:cdba::3257:965                     # A compressed version of the same address.
```

Kebijakan dual-stack IAM (IPv4 dan IPv6)

Anda dapat menggunakan kebijakan IAM untuk mengontrol akses ke Secrets Manager API dan mencegah alamat IP di luar rentang yang dikonfigurasi mengakses Secrets Manager API.

Secretsmanager. {region} .amazonaws.com titik akhir dual-stack untuk Secrets Manager API mendukung IPv6 dan IPv4.

Jika Anda perlu mendukung IPv4 dan IPv6, perbarui kebijakan pemfilteran alamat IP Anda untuk menangani alamat IPv6. Jika tidak, Anda mungkin tidak dapat terhubung ke Secrets Manager melalui IPv6.

Siapa yang harus melakukan perubahan ini?

Perubahan ini memengaruhi Anda jika Anda menggunakan pengalamatan ganda dengan kebijakan yang berisi`aws:sourceIp`. Pengalamatan ganda berarti bahwa jaringan mendukung IPv4 dan IPv6.

Jika Anda menggunakan pengalamatan ganda, perbarui kebijakan IAM Anda yang saat ini menggunakan alamat format IPv4 untuk menyertakan alamat format IPv6.

Siapa yang seharusnya tidak melakukan perubahan ini?

Perubahan ini tidak memengaruhi Anda jika Anda hanya menggunakan jaringan IPv4.

Menambahkan IPv6 ke kebijakan IAM

Kebijakan IAM menggunakan tombol `aws:SourceIp` kondisi untuk mengontrol akses dari alamat IP tertentu. Jika jaringan Anda menggunakan pengalamatan ganda (IPv4 dan IPv6), perbarui kebijakan IAM Anda untuk menyertakan rentang alamat IPv6.

Dalam `Condition` elemen kebijakan Anda, gunakan `NotIpAddress` operator `IpAddress` dan untuk kondisi alamat IP. Jangan gunakan operator string, karena mereka tidak dapat menangani berbagai format alamat IPv6 yang valid.

Contoh-contoh ini digunakan`aws:SourceIp`. Untuk VPC, gunakan `aws:VpcSourceIp` sebagai gantinya.

Berikut ini adalah Menolak akses AWS berdasarkan kebijakan referensi IP sumber dari Panduan Pengguna IAM. `NotIpAddress`Dalam `Condition` elemen untuk daftar dua rentang alamat IPv4, `192.0.2.0/24` dan`203.0.113.0/24`, yang akan ditolak akses ke API.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Deny",
        "Action": "*",
        "Resource": "*",
        "Condition": {
            "NotIpAddress": {
                "aws:SourceIp": [
                    "192.0.2.0/24",
                    "203.0.113.0/24"
                ]
            },
            "Bool": {
                "aws:ViaAWSService": "false"
            }
```

```
        }
    }
}
```

Untuk memperbarui kebijakan ini, ubah `Condition` elemen untuk menyertakan rentang `2001:DB8:1234:5678::/64` alamat IPv6 dan. `2001:cdba:3257:8593::/64`

> **Note**
>
> Jangan hapus alamat IPv4 yang ada. Mereka diperlukan untuk kompatibilitas mundur.

```
"Condition": {
                "NotIpAddress": {
                    "aws:SourceIp": [
                        "192.0.2.0/24", <<DO NOT REMOVE existing IPv4 address>>
                        "203.0.113.0/24", <<DO NOT REMOVE existing IPv4 address>>
                        "2001:DB8:1234:5678::/64", <<New IPv6 IP address>>
                        "2001:cdba:3257:8593::/64" <<New IPv6 IP address>>
                    ]
                },
                "Bool": {
                    "aws:ViaAWSService": "false"
                }
            }
```

Untuk memperbarui kebijakan ini untuk VPC, gunakan `aws:VpcSourceIp` sebagai pengganti: `aws:SourceIp`

```
"Condition": {
                "NotIpAddress": {
                    "aws:VpcSourceIp": [
                        "10.0.2.0/24", <<DO NOT REMOVE existing IPv4 address>>
                        "10.0.113.0/24", <<DO NOT REMOVE existing IPv4 address>>
                        "fc00:DB8:1234:5678::/64", <<New IPv6 IP address>>
                        "fc00:cdba:3257:8593::/64" <<New IPv6 IP address>>
                    ]
                },
                "Bool": {
                    "aws:ViaAWSService": "false"
                }
```

```
        }
```

Memverifikasi klien Anda mendukung IPv6

Jika Anda menggunakan secretsmanager. Titik akhir {region} .amazonaws.com, verifikasi bahwa Anda dapat terhubung dengannya. Langkah-langkah berikut menjelaskan cara melakukan verifikasi.

Contoh ini menggunakan Linux dan curl versi 8.6.0 dan menggunakan AWS Secrets Manager layanan yang memiliki titik akhir berkemampuan IPv6 yang terletak di titik akhir amazonaws.com.

> **Note**
>
> Secretsmanager. {region} .amazonaws.com berbeda dari konvensi penamaan dual-stack yang khas. Untuk daftar lengkap titik akhir Secrets Manager, lihatAWS Secrets Manager titik akhir.
> Ubah Wilayah AWS ke Wilayah yang sama di mana layanan Anda berada. Dalam contoh ini, kita menggunakan US East (N. Virginia) — `us-east-1` endpoint.

1. Tentukan apakah titik akhir diselesaikan dengan alamat IPv6 menggunakan perintah berikut. `dig`

   ```
   $ dig +short AAAA secretsmanager.us-east-1.amazonaws.com

   > 2600:1f18:e2f:4e05:1a8a:948e:7c08:c1c3
   ```

2. Tentukan apakah jaringan klien dapat membuat koneksi IPv6 menggunakan perintah berikut`curl`. Kode respons 404 berarti koneksi berhasil, sedangkan kode respons 0 berarti koneksi gagal.

   ```
   $ curl --ipv6 -o /dev/null --silent -w "\nremote ip: %{remote_ip}\nresponse code:
    %{response_code}\n" https://secretsmanager.us-east-1.amazonaws.com

   > remote ip: 2600:1f18:e2f:4e05:1a8a:948e:7c08:c1c3
   > response code: 404
   ```

Jika IP jarak jauh diidentifikasi dan kode respons tidak`0`, koneksi jaringan berhasil dibuat ke titik akhir menggunakan IPv6. IP jarak jauh harus berupa alamat IPv6 karena sistem operasi harus memilih protokol yang valid untuk klien.

Jika IP jarak jauh kosong atau kode responsnya`0`, jaringan klien atau jalur jaringan ke titik akhir adalah IPv4-only. Anda dapat memverifikasi konfigurasi ini dengan `curl` perintah berikut.

```
$ curl -o /dev/null --silent -w "\nremote ip: %{remote_ip}\nresponse code:
 %{response_code}\n" https://secretsmanager.us-east-1.amazonaws.com

> remote ip: 3.123.154.250
> response code: 404
```

Ketahanan di AWS Secrets Manager

AWS membangun infrastruktur global di sekitar Wilayah AWS dan Availability Zones. Wilayah AWS menyediakan beberapa Availability Zone yang terpisah secara fisik dan terisolasi, yang terhubung dengan latensi rendah, throughput tinggi, dan jaringan yang 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. Availability Zones memungkinkan Anda menjadi lebih tersedia, toleran terhadap kesalahan, dan skalabel daripada infrastruktur pusat data tunggal atau ganda tradisional.

Untuk informasi lebih lanjut tentang ketahanan dan pemulihan bencana, lihat Pilar Keandalan — AWS Well-Architected Kerangka Kerja.

Untuk informasi selengkapnya tentang Wilayah AWS dan Availability Zone, lihat Infrastruktur AWS Global.

Post-quantum TLS

Secrets Manager mendukung opsi pertukaran kunci pasca-kuantum hibrida untuk protokol enkripsi jaringan Transport Layer Security (TLS). Anda dapat menggunakan opsi TLS ini saat terhubung ke titik akhir API Secrets Manager. Kami menawarkan fitur ini sebelum algoritma pasca-kuantum distandarisasi sehingga Anda dapat mulai menguji efek protokol pertukaran kunci ini pada panggilan Secrets Manager. Fitur pertukaran kunci pasca-kuantum hibrida opsional ini setidaknya seaman enkripsi TLS yang kami gunakan saat ini dan kemungkinan akan memberikan manfaat keamanan tambahan. Namun, fitur-fitur tersebut memengaruhi latensi dan throughput dibandingkan dengan protokol pertukaran kunci klasik yang digunakan saat ini. Penyedia Kredensial AWS Beban Kerja menggunakan pertukaran kunci pasca-kuantum sebagai pertukaran ML-KEM kunci prioritas tertinggi secara default.

Untuk melindungi data yang dienkripsi hari ini terhadap potensi serangan future,AWS berpartisipasi dengan komunitas kriptografi dalam pengembangan algoritma tahan kuantum atau pasca-kuantum. Kami telah menerapkan suite cipher pertukaran kunci pasca-kuantum hibrida di titik akhir Secrets Manager. Cipher suite hibrida ini, yang menggabungkan elemen klasik dan pasca-kuantum, memastikan bahwa koneksi TLS Anda setidaknya sekuat itu dengan cipher suite klasik. Namun, karena karakteristik kinerja dan persyaratan bandwidth suite cipher hybrid berbeda dari mekanisme pertukaran kunci klasik, kami sarankan Anda mengujinya pada panggilan API Anda.

Secrets Manager mendukung PQTLS di semua Wilayah kecuali Wilayah Tiongkok.

Untuk mengkonfigurasi TLS pasca-kuantum hibrida

1. Tambahkan klien AWS Common Runtime ke dependensi Maven Anda. Sebaiknya gunakan versi terbaru yang tersedia. Misalnya, pernyataan ini menambahkan versi 2.20.0.

   ```
   <dependency>
     <groupId>software.amazon.awssdk</groupId>
     <artifactId>aws-crt-client</artifactId>
     <version>2.20.0</version>
   </dependency>
   ```

2. Tambahkan AWS SDK for Java 2.x ke project Anda dan inisialisasi. Aktifkan suite sandi pasca-kuantum hibrida pada klien HTTP Anda.

   ```
   SdkAsyncHttpClient awsCrtHttpClient = AwsCrtAsyncHttpClient.builder()
               .postQuantumTlsEnabled(true)
               .build();
   ```

3. Buat klien asinkron Secrets Manager.

   ```
   SecretsManagerAsyncClient SecretsManagerAsync = SecretsManagerAsyncClient.builder()
               .httpClient(awsCrtHttpClient)
               .build();
   ```

   Sekarang ketika Anda memanggil operasi Secrets Manager API, panggilan Anda ditransmisikan ke titik akhir Secrets Manager menggunakan TLS pasca-kuantum hibrida.

Untuk informasi lebih lanjut tentang penggunaan TLS pasca-kuantum hibrida, lihat:

- AWS SDK for Java 2.x Panduan Pengembang dan posting blog yang AWS SDK for Java 2.x dirilis.

- Memperkenalkans2n-tls, Implementasi dan Penggunaan s2n-tls TLS Open Source Baru.
- Post-QuantumKriptografi di Institut Nasional untuk Standar dan Teknologi (NIST).
- Metode Enkapsulasi Post-Quantum Kunci Hybrid (PQ KEM) untuk Transport Layer Security 1.2 (TLS).

Post-quantum TLS for Secrets Manager tersedia di semua Wilayah AWS kecuali China.

Pemecahan masalah AWS Secrets Manager

Gunakan informasi di sini untuk membantu Anda mendiagnosis dan memperbaiki masalah yang mungkin Anda temui saat bekerja dengan Secrets Manager.

Untuk masalah yang terkait dengan rotasi, lihatthe section called “Memecahkan masalah rotasi”.

Topik

- Pesan “Akses ditolak”
- “Akses ditolak” untuk kredensi keamanan sementara
- Perubahan yang saya buat tidak selalu langsung terlihat.
- Kesalahan “Koneksi diatur ulang oleh rekan”
- “Tidak dapat menghasilkan kunci data dengan kunci KMS asimetris” saat membuat rahasia
- Sesi AWS CLI atau AWS Operasi SDK tidak dapat menemukan rahasia saya dari ARN sebagian
- Rahasia ini dikelola oleh seorang AWS layanan, dan Anda harus menggunakan layanan itu untuk memperbaruinya.
- Impor modul Python gagal saat menggunakan Transform: -2024-09-16 AWS::SecretsManager
- Rahasia eksternal saya yang dikelola tidak berputar

Pesan “Akses ditolak”

Saat melakukan panggilan API seperti GetSecretValue atau CreateSecret ke Secrets Manager, Anda harus memiliki izin IAM untuk melakukan panggilan tersebut. Anda juga harus memastikan bahwa Anda melakukan panggilan API ini di AWS akun dan AWS Wilayah tempat rahasia berada. Saat Anda menggunakan konsol, konsol melakukan panggilan API yang sama atas nama Anda, jadi Anda juga harus memiliki izin IAM. Administrator dapat memberikan izin dengan melampirkan kebijakan IAM ke pengguna IAM Anda, atau ke grup tempat Anda menjadi anggota. Jika pernyataan kebijakan yang memberikan izin tersebut menyertakan syarat apapun, seperti batas waktu hari atau alamat IP, maka Anda juga harus memenuhi persyaratan tersebut ketika Anda mengirim permintaan. Untuk informasi tentang melihat atau mengubah kebijakan untuk pengguna, grup, atau peran IAM, lihat Bekerja dengan Kebijakan dalam Panduan Pengguna IAM. Untuk informasi tentang izin yang diperlukan untuk Secrets Manager, lihatthe section called “Kontrol autentikasi dan akses”.

Jika Anda menandatangani permintaan API secara manual, tanpa menggunakan AWS SDK, pastikan Anda menandatangani permintaan dengan benar.

“Akses ditolak” untuk kredensi keamanan sementara

Verifikasi pengguna IAM atau peran yang Anda gunakan untuk membuat permintaan memiliki izin yang benar. Izin untuk kredensi keamanan sementara berasal dari pengguna atau peran IAM. Ini berarti izin terbatas pada yang diberikan kepada pengguna atau peran IAM. Untuk informasi lebih lanjut tentang bagaiman izin kredensial keamanan sementara ditentukan, lihat Mengontrol Izin untuk Kredensial Keamanan Sementara dalam Panduan Pengguna IAM.

Verifikasi bahwa permintaan Anda ditandatangani dengan benar dan permintaan tersebut terbentuk dengan baik. Untuk detailnya, lihat dokumentasi toolkit untuk SDK yang Anda pilih, atau Menggunakan Kredensial Keamanan Sementara untuk Meminta Akses ke AWS Sumber Daya di Panduan Pengguna IAM.

Verifikasikan bahwa kredensial keamanan sementara Anda belum kedaluwarsa. Untuk informasi lebih lanjut, lihat Meminta Kredensial Keamanan Sementara dalam Panduan Pengguna IAM.

Untuk informasi tentang izin yang diperlukan untuk Secrets Manager, lihatthe section called “Kontrol autentikasi dan akses”.

Perubahan yang saya buat tidak selalu langsung terlihat.

Secrets Manager menggunakan model komputasi terdistribusi yang disebut konsistensi akhirnya. Setiap perubahan yang Anda buat di Secrets Manager (atau AWS layanan lainnya) membutuhkan waktu untuk terlihat dari semua titik akhir yang mungkin. Beberapa hasil penundaan dari waktu yang diperlukan untuk mengirim data dari server ke server, dari zona replikasi ke zona replikasi, dan dari wilayah ke wilayah di seluruh dunia. Secrets Manager juga menggunakan caching untuk meningkatkan kinerja, tetapi dalam beberapa kasus ini dapat menambah waktu. Perubahan mungkin tidak terlihat sampai waktu data yang disimpan di-cache sebelumnya habis.

Rancang aplikasi global Anda untuk memperhitungkan potensi penundaan ini. Juga, pastikan bahwa mereka bekerja seperti yang diharapkan, bahkan ketika perubahan yang dibuat di satu lokasi tidak langsung terlihat di lokasi lain.

Untuk informasi selengkapnya tentang bagaimana beberapa AWS layanan lain dipengaruhi oleh konsistensi akhirnya, lihat:

- Mengelola konsistensi data dalam Panduan Pengembang Database Amazon Redshift
- Model Konsistensi Data Amazon S3 di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon

- Memastikan Konsistensi Saat Menggunakan Amazon S3 dan Amazon EMR untuk Alur Kerja ETL di Blog Big Data AWS
- Konsistensi Akhirnya Amazon EC2 dalam Referensi API Amazon EC2

Kesalahan “Koneksi diatur ulang oleh rekan”

Ketika Anda memanggil Secrets Manager, Anda mungkin sesekali menerima koneksi reset oleh kesalahan rekan. Ini terjadi ketika server menutup koneksi TCP yang klien Anda gunakan atau coba gunakan. Untuk mengelola kapasitas dan menjaga ketersediaan, AWS layanan, termasuk Secrets Manager, dapat menutup koneksi kapan saja.

Ini paling sering memengaruhi koneksi keep-alive idle yang digunakan kembali aplikasi Anda, tetapi juga dapat memengaruhi koneksi yang baru dibuat. Reset sesekali adalah bagian normal dari manajemen siklus hidup koneksi. Mereka tidak menunjukkan masalah dengan rahasia Anda atau akun Anda.

AWS SDK mengelola kumpulan koneksi dan mencoba lagi kesalahan ini pada koneksi baru secara default, sehingga Anda biasanya tidak perlu mengambil tindakan apa pun. Jika Anda mengalami kesalahan ini, lakukan hal berikut:

- Verifikasi bahwa Anda menggunakan versi AWS SDK saat ini atau AWS CLI. Pastikan Anda belum mematikan perilaku coba ulang default. Untuk informasi selengkapnya, lihat Mencoba lagi perilaku di AWS SDK dan Panduan Referensi Alat.
- Jika Anda mengelola koneksi HTTP secara langsung alih-alih menggunakan AWS SDK, coba lagi permintaan pada koneksi baru. Operasi baca seperti `GetSecretValue` idempoten dan aman untuk dicoba lagi.
- Cache nilai rahasia pada klien alih-alih memanggil Secrets Manager di setiap penggunaan. Caching mengurangi jumlah koneksi yang dibuka aplikasi Anda dan meningkatkan ketahanan terhadap kesalahan jaringan sementara. Untuk informasi selengkapnya, lihat Gunakan caching untuk mengambil rahasia.

Jika kesalahan ini bertahan selama periode yang berkelanjutan, penyebabnya kemungkinan adalah sesuatu selain siklus koneksi rutin. Untuk mengidentifikasi penyebabnya, periksa Dasbor AWS Kesehatan untuk acara Secrets Manager di AWS Wilayah Anda. Juga verifikasi jalur jaringan Anda ke titik akhir layanan, termasuk titik akhir VPC, proxy, firewall, atau konfigurasi DNS apa pun. Jika masalah berlanjut, hubungi AWS Support.

“Tidak dapat menghasilkan kunci data dengan kunci KMS asimetris” saat membuat rahasia

Secrets Manager menggunakan kunci KMS enkripsi simetris yang terkait dengan rahasia untuk menghasilkan kunci data untuk setiap nilai rahasia. Anda tidak dapat menggunakan tombol KMS asimetris. Verifikasi Anda menggunakan kunci KMS enkripsi simetris alih-alih kunci KMS asimetris. Untuk petunjuk, lihat Mengidentifikasi kunci KMS asimetris.

Sesi AWS CLI atau AWS Operasi SDK tidak dapat menemukan rahasia saya dari ARN sebagian

Dalam banyak kasus, Secrets Manager dapat menemukan rahasia Anda dari bagian ARN daripada ARN penuh. Namun, jika nama rahasia Anda berakhir dengan tanda hubung diikuti oleh enam karakter, Secrets Manager mungkin tidak dapat menemukan rahasia hanya dari sebagian ARN. Sebagai gantinya, kami sarankan Anda menggunakan ARN lengkap atau nama rahasianya.

Lebih detail

Secrets Manager mencakup enam karakter acak di akhir nama rahasia untuk membantu memastikan bahwa ARN rahasia itu unik. Jika rahasia asli dihapus, dan kemudian rahasia baru dibuat dengan nama yang sama, kedua rahasia memiliki ARN yang berbeda karena karakter ini. Pengguna dengan akses ke rahasia lama tidak secara otomatis mendapatkan akses ke rahasia baru karena ARN berbeda.

Secrets Manager membangun ARN untuk rahasia dengan Region, akun, nama rahasia, dan kemudian tanda hubung dan enam karakter lagi, sebagai berikut:

```
arn:aws:secretsmanager:us-east-2:111122223333:secret:SecretName-abcdef
```

Jika nama rahasia Anda diakhiri dengan tanda hubung dan enam karakter, hanya menggunakan sebagian dari ARN dapat muncul ke Secrets Manager seolah-olah Anda menentukan ARN lengkap. Misalnya, Anda mungkin memiliki rahasia bernama `MySecret-abcdef` ARN

```
arn:aws:secretsmanager:us-east-2:111122223333:secret:MySecret-abcdef-nutBrk
```

Jika Anda memanggil operasi berikut, yang hanya menggunakan bagian dari ARN rahasia, maka Secrets Manager mungkin tidak menemukan rahasianya.

```
$ aws secretsmanager describe-secret --secret-id arn:aws:secretsmanager:us-
east-2:111122223333:secret:MySecret-abcdef
```

Rahasia ini dikelola oleh seorang AWS layanan, dan Anda harus menggunakan layanan itu untuk memperbaruinya.

Jika Anda menemukan pesan ini saat mencoba memodifikasi rahasia, rahasia hanya dapat diperbarui dengan menggunakan layanan pengelolaan yang tercantum dalam pesan. Untuk informasi selengkapnya, lihat Rahasia dikelola oleh layanan lain.

Untuk menentukan siapa yang mengelola rahasia, Anda dapat meninjau nama rahasia. Rahasia yang dikelola oleh layanan lain diawali dengan ID layanan tersebut. Atau, di AWS CLI, panggil deskripsikan-rahasia, dan kemudian tinjau bidangnya. `OwningService`

Impor modul Python gagal saat menggunakan **`Transform`**: -2024-09-16 AWS::SecretsManager

Jika Anda menggunakan Transform: `AWS::SecretsManager-2024-09-16` dan mengalami kegagalan impor modul Python saat fungsi Lambda rotasi Anda berjalan, masalah kemungkinan disebabkan oleh nilai yang tidak kompatibel. `Runtime` Dengan versi transformasi ini, AWS CloudFormation mengelola versi runtime, kode, dan file objek bersama untuk Anda. Anda tidak perlu mengelolanya sendiri.

Rahasia eksternal saya yang dikelola tidak berputar

Jika a Rahasia yang dikelola oleh aplikasi pihak ketiga tidak berputar seperti yang diharapkan, Secrets Manager mencatat rincian dalam suatu `RotationFailed` peristiwa di AWS CloudTrail. Acara ini mencakup informasi spesifik tentang mengapa rotasi gagal. Tinjau `RotationFailed` peristiwa terbaru untuk rahasia untuk mengidentifikasi penyebabnya. Untuk informasi selengkapnya tentang melihat acara Secrets Manager di CloudTrail, lihatthe section called “Masuk dengan AWS CloudTrail ”.

Berikut ini adalah penyebab umum kegagalan rotasi untuk rahasia eksternal yang dikelola:

- Kredensi tidak valid. Kredensil yang disimpan dalam rahasia, atau di rahasia admin jika Anda menggunakannya, tidak lagi valid atau tidak memiliki izin untuk memutar rahasia di aplikasi mitra.

Verifikasi bahwa nilai rahasia berisi bidang yang benar dalam format yang dibutuhkan mitra integrasi, dan kredensialnya masih aktif di aplikasi mitra. Untuk bidang yang diperlukan untuk setiap mitra, lihatMitra Integrasi.

- Akses jaringan yang diblokir ke sumber daya eksternal. Aplikasi mitra memblokir lalu lintas rotasi rahasia eksternal yang dikelola Secrets Manager. Jika sumber daya eksternal Anda membatasi akses masuk dengan alamat IP, Anda harus mengizinkan daftar rentang IP tempat panggilan rotasi berasal. Tambahkan daftar awalan AWS-managed`com.amazonaws.`*`region`*`.secretsmanager-managed-external-secrets`, di mana *`region`* AWS Wilayah tempat rahasia Anda ada, ke aturan ingress Anda. Menggunakan daftar awalan terkelola memastikan aturan masuk Anda secara otomatis tetap terkini saat rentang IP yang mendasarinya berubah. Untuk informasi selengkapnya, lihat daftar awalan AWS-terkelola di Panduan Pengguna Amazon VPC.

  Jika Anda lebih suka mereferensikan daftar awalan secara terprogram, Anda dapat menggunakan GetManagedPrefixListEntriesAPI untuk membuat daftar IP yang perlu diijinkan. Anda harus me-refresh aturan ingress Anda secara berkala dengan daftar awalan ini agar tetap terkini.

Untuk informasi selengkapnya tentang rotasi rahasia eksternal terkelola, lihatRotate Secrets Manager mengelola rahasia eksternal. Untuk detail keamanan dan izin, termasuk daftar awalan terkelola, lihat. Keamanan dan izin

AWS Secrets Manager kuota

Secrets Manager read API memiliki kuota TPS yang tinggi, dan API control plane yang jarang disebut memiliki kuota TPS yang lebih rendah. Kami menyarankan Anda menghindari menelepon `PutSecretValue` atau dengan `UpdateSecret` kecepatan berkelanjutan lebih dari sekali setiap 10 menit. Saat Anda `UpdateSecret` menelepon `PutSecretValue` atau memperbarui nilai rahasia, Secrets Manager membuat versi baru dari rahasia tersebut. Secrets Manager menghapus versi yang tidak berlabel ketika ada lebih dari 100, tetapi tidak menghapus versi yang dibuat kurang dari 24 jam yang lalu. Jika Anda memperbarui nilai rahasia lebih dari sekali setiap 10 menit, Anda membuat lebih banyak versi daripada yang dihapus Secrets Manager, dan Anda akan mencapai kuota untuk versi rahasia.

Anda dapat mengoperasikan beberapa wilayah di akun Anda, dan setiap kuota khusus untuk setiap wilayah.

Ketika aplikasi dalam satu Akun AWS menggunakan rahasia yang dimiliki oleh akun yang berbeda, itu dikenal sebagai permintaan lintas akun. Untuk permintaan lintas akun, Secrets Manager membatasi akun identitas yang membuat permintaan, bukan akun yang memiliki rahasia. Misalnya, jika identitas dari akun A menggunakan rahasia di akun B, penggunaan rahasia hanya berlaku untuk kuota di akun A.

Kuota Secrets Manager

Nama	Default	Dapat disesuaikan	Deskripsi
Tingkat gabungan permintaan DeleteResourcePolicy GetResourcePolicy, PutResourcePolicy,, dan ValidateResourcePolicy API	Setiap Wilayah yang didukung: 50 per detik	Tidak	Transaksi maksimum per detik untuk DeleteResourcePolicy, GetResourcePolicy, PutResourcePolicy, dan permintaan ValidateResourcePolicy API digabungkan.

Nama	Default	Dapat disesuaikan	Deskripsi
Tingkat gabungan permintaan PutSecretValue, RemoveRegionsFromReplication ReplicateSecretToRegion, StopReplicationToReplica, UpdateSecret,, dan UpdateSecretVersionStage API	Setiap Wilayah yang didukung: 50 per detik	Tidak	Transaksi maksimum per detik untuk PutSecretValue, RemoveRegionsFromReplication,, ReplicateSecretToRegion, StopReplicationToReplica, UpdateSecret, dan permintaan UpdateSecretVersionStage API digabungkan.
Tingkat gabungan permintaan RestoreSecret API	Setiap Wilayah yang didukung: 50 per detik	Tidak	Transaksi maksimum per detik untuk permintaan RestoreSecret API.
Tingkat gabungan permintaan RotateSecret dan CancelRotateSecret API	Setiap Wilayah yang didukung: 50 per detik	Tidak	Transaksi maksimum per detik untuk RotateSecret dan permintaan CancelRotateSecret API digabungkan.
Tingkat gabungan permintaan TagResource dan UntagResource API	Setiap Wilayah yang didukung: 50 per detik	Tidak	Transaksi maksimum per detik untuk TagResource dan permintaan UntagResource API digabungkan.
Tingkat permintaan BatchGetSecretValue API	Setiap Wilayah yang didukung: 100 per detik	Tidak	Transaksi maksimum per detik untuk permintaan BatchGetSecretValue API.

Nama	Default	Dapat disesu an	Deskripsi
Tingkat permintaan CreateSecret API	Setiap Wilayah yang didukung: 50 per detik	Tidak	Transaksi maksimum per detik untuk permintaan CreateSecret API.
Tingkat permintaan DeleteSecret API	Setiap Wilayah yang didukung: 50 per detik	Tidak	Transaksi maksimum per detik untuk permintaan DeleteSecret API.
Tingkat permintaan DescribeSecret API	Setiap Wilayah yang didukung: 40.000 per detik	Tidak	Transaksi maksimum per detik untuk permintaan DescribeSecret API.
Tingkat permintaan GetRandom Password API	Setiap Wilayah yang didukung: 50 per detik	Tidak	Transaksi maksimum per detik untuk permintaan GetRandomPassword API.
Tingkat permintaan GetSecretValue API	Setiap Wilayah yang didukung: 10.000 per detik	Tidak	Transaksi maksimum per detik untuk permintaan GetSecretValue API.
Tingkat permintaan ListSecretVersionI ds API	Setiap Wilayah yang didukung: 50 per detik	Tidak	Transaksi maksimum per detik untuk permintaan ListSecretVersionIds API.
Tingkat permintaan ListSecrets API	Setiap Wilayah yang didukung: 100 per detik	Tidak	Transaksi maksimum per detik untuk permintaan ListSecrets API.
Resource-based panjang kebijakan	Setiap Wilayah yang didukung: 20.480	Tidak	Jumlah maksimum karakter dalam kebijakan izin berbasis sumber daya yang dilampirkan pada rahasia.

Nama	Default	Dapat disesu an	Deskripsi
Ukuran nilai rahasia	Setiap Wilayah yang didukung: 65.536 Bytes	Tidak	Ukuran maksimum nilai rahasia terenkripsi. Jika nilai rahasia adalah string, maka ini adalah jumlah karakter yang diizinkan dalam nilai rahasia.
Rahasia	Setiap Wilayah yang didukung: 500.000	Tidak	Jumlah maksimum rahasia di setiap AWS Wilayah AWS akun ini.
Label pementasan yang dilampirkan di semua versi rahasia	Setiap Wilayah yang didukung: 20	Tidak	Jumlah maksimum label pementasan yang dilampirkan di semua versi rahasia.
Versi per rahasia	Setiap Wilayah yang didukung: 100	Tidak	Jumlah maksimum versi rahasia.

Tambahkan percobaan ulang ke aplikasi Anda

AWS Klien Anda mungkin melihat panggilan ke Secrets Manager gagal karena masalah yang tidak terduga. Atau panggilan mungkin gagal karena pembatasan tarif dari Secrets Manager. Jika Anda melebihi kuota permintaan API, Secrets Manager membatasi permintaan tersebut. Ini menolak permintaan yang valid dan mengembalikan throttling kesalahan. Untuk kedua jenis kegagalan, kami sarankan Anda mencoba lagi panggilan setelah masa tunggu singkat. Ini disebut strategi backoff dan coba lagi.

Jika mengalami kesalahan berikut, Anda mungkin ingin menambahkan percobaan ulang ke kode aplikasi Anda:

Kesalahan sementara dan pengecualian

- `RequestTimeout`
- `RequestTimeoutException`
- `PriorRequestNotComplete`
- `ConnectionError`
- `HTTPClientError`

Service-side membatasi dan membatasi kesalahan dan pengecualian

- `Throttling`
- `ThrottlingException`
- `ThrottledException`
- `RequestThrottledException`
- `TooManyRequestsException`
- `ProvisionedThroughputExceededException`
- `TransactionInProgressException`
- `RequestLimitExceeded`
- `BandwidthLimitExceeded`
- `LimitExceededException`
- `RequestThrottled`
- `SlowDown`

Untuk informasi selengkapnya, serta kode contoh, tentang percobaan ulang, backoff eksponensial, dan jitter, lihat sumber daya berikut:

- Backoff dan Jitter Eksponensial
- Batas waktu, percobaan ulang, dan backoff dengan jitter
- Kesalahan mencoba ulang dan backoff eksponensial di. AWS

Riwayat dokumen

Tabel berikut menjelaskan perubahan penting pada dokumentasi sejak rilis terakhir AWS Secrets Manager. Untuk notifikasi tentang pembaruan dokumentasi ini, Anda dapat berlangganan ke umpan RSS.

Perubahan	Deskripsi	Tanggal
Perbarui ke kebijakan AWS terkelola	Kebijakan `AWSSecretsManagerClientReadOnlyAccess` terkelola sekarang menyertakan `BatchGetSecretValue` dan `ListSecrets` izin. Untuk selengkapnya, lihat Pembaruan Manajer Rahasia untuk kebijakan AWS terkelola.	Juni 2, 2026
Kebijakan ter AWS kelola baru	Secrets Manager telah merilis kebijakan terkelola baru `AWSSecretsManagerClientReadOnlyAccess` yang menyediakan akses baca-saja ke rahasia untuk aplikasi klien. Untuk selengkapnya, lihat Pembaruan Manajer Rahasia untuk kebijakan AWS terkelola.	5 November 2025
Menambahkan dukungan untuk tag alokasi biaya	Secrets Manager sekarang mendukung tag alokasi biaya, memungkinkan pelanggan untuk mengkategorikan dan melacak biaya berdasarkan departemen, tim, atau aplikasi. Untuk informasi selengkap	Mei 27, 2025

	nya, lihat Menggunakan tag alokasi biaya dengan AWS Secrets Manager.	
Menambahkan dukungan IPv6 dan dual-stack	Secrets Manager sekarang mendukung titik akhir dual-stack. Lihat akses IPv4 dan IPv6 untuk informasi lebih lanjut.	Desember 20, 2024
Pengelola Rahasia berubah menjadi kebijakan AWS terkelola	Kebijakan ter `SecretsManagerReadWrite` kelola sekarang menyertakan `redshift-serverless` izin. Untuk informasi selengkapnya, lihat kebijakan AWS terkelola untuk AWS Secrets Manager	Maret 12, 2024

Pembaruan lebih awal

Tabel berikut menjelaskan perubahan penting dalam setiap rilis Panduan AWS Secrets Manager Pengguna sebelum Februari 2024.

Ubah	Deskripsi	Date
Ketersediaan umum	Ini adalah rilis publik awal dari Secrets Manager.	4 April 2018

Terjemahan disediakan oleh mesin penerjemah. Jika konten terjemahan yang diberikan bertentangan dengan versi bahasa Inggris aslinya, utamakan versi bahasa Inggris.