



Panduan Pengguna

# AWS Secrets Manager



# AWS Secrets Manager: Panduan Pengguna

Copyright © 2024 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

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

---

# Table of Contents

Apa itu Secrets Manager? .....	1
Memulai Secrets Manager .....	1
Kepatuhan dengan standar .....	2
Harga .....	2
Akses Secrets Manager .....	3
Konsol Secrets Manager .....	3
Alat baris perintah .....	3
AWS SDK .....	4
API Kueri HTTPS .....	4
Titik akhir Secrets Manager .....	5
Apa yang ada dalam rahasia .....	10
Metadata .....	10
Versi rahasia .....	11
Tutorial .....	13
CodeGuru Peninjau Amazon .....	13
Ganti rahasia hardcoded .....	13
Langkah 1: Buat rahasia .....	14
Langkah 2: Perbarui kode Anda .....	16
Langkah 3: Perbarui rahasianya .....	17
Langkah selanjutnya .....	17
Ganti kredensial DB hardcoded .....	18
Langkah 1: Buat Rahasianya .....	18
Langkah 2: Perbarui kode Anda .....	20
Langkah 3: Putar rahasianya .....	20
Langkah selanjutnya .....	21
Rotasi pengguna bergantian .....	22
Izin .....	23
Prasyarat .....	23
Langkah 1: Buat pengguna database Amazon RDS .....	26
Langkah 2: Buat rahasia untuk kredensi pengguna .....	29
Langkah 3: Uji rahasia yang diputar .....	30
Langkah 4: Bersihkan Sumber Daya .....	31
Langkah selanjutnya .....	31
Rotasi pengguna tunggal .....	31

Izin .....	32
Prasyarat .....	32
Langkah 1: Buat pengguna database Amazon RDS .....	32
Langkah 2: Buat rahasia untuk kredensi pengguna database .....	33
Langkah 3: Uji kata sandi yang diputar .....	34
Langkah 4: Bersihkan Sumber Daya .....	35
Langkah selanjutnya .....	35
Kontrol autentikasi dan akses .....	36
Izin administrator Secrets Manager .....	36
Izin untuk mengakses rahasia .....	36
Izin untuk fungsi rotasi Lambda .....	37
Izin untuk kunci enkripsi .....	37
Izin untuk replikasi .....	37
Melampirkan kebijakan izin ke identitas .....	37
Lampirkan kebijakan izin ke rahasia .....	38
AWS CLI .....	39
AWS SDK .....	40
AWS kebijakan terkelola .....	40
SecretsManagerReadWrite .....	41
Pembaruan kebijakan .....	43
Tentukan siapa yang memiliki izin untuk rahasia Anda .....	44
Akses lintas akun .....	45
Akses di tempat .....	47
Contoh kebijakan izin .....	48
Contoh: Izin untuk mengambil nilai rahasia individu .....	49
Contoh: Izin untuk membaca dan menggambarkan rahasia individu .....	50
Contoh: Izin untuk mengambil sekelompok nilai rahasia dalam batch .....	51
Contoh: Wildcard .....	52
Contoh: Izin untuk membuat rahasia .....	53
Contoh: Tolak AWS KMS kunci tertentu untuk mengenkripsi rahasia .....	54
Contoh: Izin dan VPC .....	55
Contoh: Kontrol akses ke rahasia menggunakan tag .....	57
Contoh: Batasi akses ke identitas dengan tag yang cocok dengan tag rahasia .....	58
Contoh: Prinsipal layanan .....	59
Referensi izin .....	60
Tindakan Secrets Manager .....	61

Sumber daya Secrets Manager .....	86
Kunci syarat .....	86
BlockPublicPolicykondisi .....	89
Kondisi alamat IP .....	90
Kondisi titik akhir VPC .....	90
Buat dan kelola rahasia .....	91
Buat rahasia database .....	91
AWS CLI .....	94
AWS SDK .....	94
Struktur JSON dari sebuah rahasia .....	94
Struktur rahasia Amazon RDS Db2 .....	95
Struktur rahasia Amazon RDS MariaDB .....	96
Amazon RDS dan Amazon Aurora MySQL struktur rahasia .....	96
Struktur rahasia Amazon RDS Oracle .....	97
Amazon RDS dan Amazon Aurora PostgreSQL struktur rahasia .....	97
Amazon RDS Microsoft SQLServer struktur rahasia .....	98
Struktur rahasia Amazon DocumentDB .....	98
Struktur rahasia Amazon Redshift .....	99
Amazon Redshift Struktur rahasia tanpa server .....	100
Struktur ElastiCache rahasia Amazon .....	100
Struktur rahasia Direktori Aktif .....	101
Buat rahasia .....	102
AWS CLI .....	105
AWS SDK .....	105
Perbarui nilai rahasia .....	105
AWS CLI .....	106
AWS SDK .....	107
Buat kata sandi dengan Secrets Manager .....	107
Kembalikan rahasia ke versi sebelumnya .....	107
Ubah kunci enkripsi untuk rahasia .....	108
AWS CLI .....	109
Memodifikasi rahasia .....	110
AWS CLI .....	111
AWS SDK .....	112
Temukan rahasia .....	112
AWS CLI .....	113

AWS SDK .....	114
Hapus rahasia .....	114
AWS CLI .....	115
AWS SDK .....	116
Kembalikan rahasia .....	116
AWS CLI .....	117
AWS SDK .....	117
Rahasia tag .....	118
AWS CLI .....	118
AWS SDK .....	119
Replikasi rahasia di seluruh Wilayah .....	120
AWS CLI .....	121
AWS SDK .....	122
Promosikan rahasia replika ke rahasia mandiri .....	122
AWS CLI .....	123
AWS SDK .....	123
Mencegah replikasi .....	123
Pecahkan masalah replikasi .....	125
Rahasia dengan nama yang sama ada di Wilayah yang dipilih .....	125
Tidak ada izin yang tersedia pada kunci KMS untuk menyelesaikan replikasi .....	125
Kunci KMS dinonaktifkan atau tidak ditemukan .....	125
Anda belum mengaktifkan Wilayah tempat replikasi terjadi .....	126
Dapatkan rahasia .....	127
Java .....	127
Java dengan caching sisi klien .....	128
Koneksi JDBC dengan kredensial secara rahasia .....	135
AWS SDK Jawa .....	144
Python .....	146
Python dengan caching sisi klien .....	147
SDK Python AWS .....	153
Dapatkan sejumlah nilai rahasia .....	154
.NET .....	156
.NET dengan caching sisi klien .....	156
.NET AWS SDK .....	163
Go .....	165
Pergi dengan caching sisi klien .....	166

Pergi AWS SDK .....	170
C++ .....	171
JavaScript .....	172
Kotlin .....	174
PHP .....	174
Ruby .....	175
Rust .....	176
AWS CLI .....	177
Dapatkan sekelompok rahasia dalam batch menggunakan AWS CLI .....	177
AWS konsol .....	178
AWS Batch .....	178
AWS CloudFormation .....	179
Amazon EKS .....	180
Langkah 1: Siapkan kontrol akses .....	181
Langkah 2: Instal dan konfigurasi ASACP .....	182
Langkah 3: Identifikasi rahasia mana yang akan dipasang .....	183
Langkah 4: Pasang rahasia sebagai file di pod Amazon EKS .....	186
Pemecahan Masalah .....	186
SecretProviderClass .....	187
GitHub Lowongan .....	190
Prasyarat .....	190
Penggunaan .....	191
Penamaan variabel lingkungan .....	192
Contoh .....	193
AWS IoT Greengrass .....	196
AWS Lambda .....	196
Variabel-variabel lingkungan .....	199
Penyimpanan Parameter .....	200
Putar rahasia .....	202
Rotasi terkelola .....	202
Rotasi oleh fungsi Lambda .....	204
Rotasi otomatis untuk rahasia database (konsol) .....	205
Rotasi otomatis untuk rahasia non-database (konsol) .....	208
Rotasi otomatis (AWS CLI) .....	213
Strategi rotasi fungsi Lambda .....	216
Fungsi rotasi Lambda .....	218

Templat fungsi rotasi .....	222
Izin untuk rotasi .....	229
Akses jaringan untuk fungsi rotasi Lambda .....	233
Memecahkan masalah rotasi .....	234
Putar rahasia segera .....	243
AWS CLI .....	243
Jadwal rotasi .....	243
Ekspresi rate .....	244
Ekspresi Cron .....	244
Temukan rahasia yang tidak diputar .....	250
Batalkan rotasi otomatis .....	251
Rahasia yang dikelola .....	252
Layanan yang menggunakan rahasia .....	253
App Runner .....	255
App2Container AWS .....	255
AWS AppConfig .....	255
Amazon AppFlow .....	256
AWS AppSync .....	256
Amazon Athena .....	256
Amazon Aurora .....	256
AWS CodeBuild .....	257
Amazon Data Firehose .....	257
AWS DataSync .....	257
Amazon DataZone .....	258
AWS Direct Connect .....	258
AWS Directory Service .....	258
Amazon DocumentDB .....	259
AWS Elastic Beanstalk .....	259
Amazon Elastic Container Registry .....	259
Amazon Elastic Container Service .....	260
Amazon ElastiCache .....	261
AWS Elemental Live .....	261
AWS Elemental MediaConnect .....	261
AWS Elemental MediaConvert .....	262
AWS Elemental MediaLive .....	262
AWS Elemental MediaPackage .....	262



AWS Elemental MediaTailor .....	262
Amazon EMR .....	263
EMR pada EC2 .....	263
EMR Tanpa Server .....	263
Amazon EventBridge .....	264
Amazon FSx .....	264
AWS Glue DataBrew .....	264
AWS Glue Studio .....	265
AWS IoT SiteWise .....	265
Amazon Kendra .....	265
Amazon Kinesis Video Streams .....	265
AWS Launch Wizard .....	266
Amazon Lookout for Metrics .....	266
Amazon Managed Grafana .....	266
AWS Managed Services .....	267
Amazon Managed Streaming untuk Apache Kafka .....	267
Amazon Managed Workflows for Apache Airflow (MWAA) .....	267
AWS Marketplace .....	268
AWS Migration Hub .....	268
AWS Panorama .....	268
AWS ParallelCluster .....	269
Amazon Q .....	269
AWS OpsWorks for Chef Automate .....	269
Amazon QuickSight .....	269
Amazon RDS .....	270
Amazon Redshift .....	270
Editor kueri Amazon Redshift v2 .....	271
Amazon SageMaker .....	271
AWS SCT .....	272
AWS Toolkit for JetBrains .....	272
AWS Transfer Family .....	272
AWS Wickr .....	273
Titik akhir VPC .....	274
Subnet bersama .....	275
AWS CloudFormation .....	276
Buat rahasia .....	276

JSON .....	277
YAML .....	277
Buat rahasia dengan kredensial Amazon RDS dengan rotasi otomatis .....	278
Buat rahasia dengan kredensi Amazon Redshift .....	278
Buat rahasia dengan kredensi Amazon DocumentDB .....	278
JSON .....	279
YAML .....	283
Bagaimana Secrets Manager menggunakan AWS CloudFormation .....	285
AWS CDK .....	286
Memantau rahasia .....	287
Log dengan AWS CloudTrail .....	287
AWS CLI .....	288
CloudTrail entri .....	288
Monitor dengan CloudWatch .....	293
CloudWatch alarm .....	294
Acara Match Secrets Manager dengan EventBridge .....	295
Cocokkan semua perubahan dengan rahasia tertentu .....	295
Cocokkan acara saat nilai rahasia berputar .....	296
Memantau rahasia yang dijadwalkan untuk dihapus .....	296
Langkah 1: Konfigurasi pengiriman file CloudTrail log ke CloudWatch Log .....	297
Langkah 2: Buat CloudWatch alarm .....	297
Langkah 3: Uji CloudWatch alarm .....	299
Pantau rahasia untuk kepatuhan .....	299
Biaya Monitor Secrets Manager .....	300
Validasi kepatuhan .....	301
Standar kepatuhan .....	301
Keamanan di Secrets Manager .....	304
Mengurangi risiko menggunakan AWS CLI untuk menyimpan rahasia Anda AWS Secrets Manager .....	304
Perlindungan data di Secrets Manager .....	307
Enkripsi saat tidak aktif .....	308
Enkripsi dalam transit .....	308
Privasi lalu lintas antar jaringan .....	308
Pengelolaan kunci enkripsi .....	309
Enkripsi rahasia dan dekripsi .....	309
Memilih AWS KMS kunci .....	310

Apa yang dienkripsi? .....	311
Proses enkripsi dan dekripsi .....	311
Izin untuk kunci KMS .....	312
Bagaimana Secrets Manager menggunakan kunci KMS Anda .....	312
Kebijakan utama dari Kunci yang dikelola AWS (aws/secretsmanager) .....	314
Konteks enkripsi Secrets Manager .....	316
Memantau interaksi Secrets Manager dengan AWS KMS .....	318
Keamanan infrastruktur .....	322
Ketangguhan .....	323
TLS pasca-kuantum .....	323
Pemecahan Masalah .....	326
Pesan “Akses ditolak” .....	326
“Akses ditolak” untuk kredensial keamanan sementara .....	326
Perubahan yang saya buat tidak selalu langsung terlihat. ....	327
“Tidak dapat menghasilkan kunci data dengan kunci KMS asimetris” saat membuat rahasia ...	328
Operasi AWS CLI atau AWS SDK tidak dapat menemukan rahasia saya dari ARN sebagian ...	328
Rahasia ini dikelola oleh AWS layanan, dan Anda harus menggunakan layanan itu untuk memperbaruinya. ....	329
Quotas .....	330
Kuota Secrets Manager .....	330
Tambahkan percobaan ulang ke aplikasi Anda .....	333
Riwayat dokumen .....	335
Pembaruan sebelumnya .....	335
.....	cccxxxvi

# Apa itu AWS Secrets Manager?

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

Secrets Manager membantu Anda meningkatkan postur keamanan Anda, karena Anda tidak lagi memerlukan kredensi hard-code dalam kode sumber aplikasi. Menyimpan kredensi di Secrets Manager membantu menghindari kemungkinan kompromi oleh siapa saja yang dapat memeriksa aplikasi atau komponen Anda. Anda mengganti kredensi 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 kredensialnya tidak lagi disimpan bersama aplikasi, kredensi yang berputar tidak lagi memerlukan pembaruan aplikasi Anda dan menerapkan perubahan ke klien aplikasi.

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

- AWS kredensi — Kami merekomendasikan [AWS Identity and Access Management](#)
- Kunci enkripsi — Kami merekomendasikan [AWS Key Management Service](#).
- Kunci SSH - Kami merekomendasikan [Amazon EC2 Instance Connect](#).
- Kunci pribadi dan sertifikat — Kami merekomendasikan [AWS Certificate Manager](#).

## Memulai Secrets Manager

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

- [the section called “Ganti rahasia hardcode ”](#)
- [the section called “Ganti kredensial DB hardcode ”](#)
- [the section called “Rotasi pengguna bergantian”](#)
- [the section called “Rotasi pengguna tunggal”](#)

Tugas lain yang dapat Anda lakukan dengan rahasia:

- [Buat dan kelola rahasia](#)
- [Kontrol akses ke rahasia Anda](#)
- [Dapatkan rahasia](#)
- [Putar 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, lihat [the section called “Biaya Monitor Secrets Manager”](#).

Anda dapat menggunakan Secrets Manager Kunci yang dikelola AWS `aws/secretsmanager` yang dibuat untuk mengenkripsi rahasia Anda secara gratis. Jika Anda membuat kunci KMS Anda sendiri untuk mengenkripsi rahasia Anda, AWS menagih Anda dengan tarif saat ini AWS KMS. Untuk informasi selengkapnya, silakan lihat [Harga AWS Key Management Service](#).

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

Jika Anda mengaktifkan AWS CloudTrail di akun, Anda bisa mendapatkan log panggilan API yang dikirimkan Secrets Manager. Secrets Manager mencatat semua peristiwa sebagai acara 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 lebih lanjut, lihat [Harga AWS CloudTrail](#).

# Akses AWS Secrets Manager

Anda dapat bekerja dengan Secrets Manager dengan salah satu cara berikut:

- [Konsol Secrets Manager](#)
- [Alat baris perintah](#)
- [AWS SDK](#)
- [API Kueri HTTPS](#)
- [AWS Secrets Manager titik akhir](#)

## Konsol Secrets Manager

Anda dapat mengelola rahasia Anda menggunakan [konsol Secrets Manager](#) berbasis browser dan melakukan hampir semua tugas yang terkait dengan rahasia Anda dengan menggunakan konsol.

## Alat baris perintah

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

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”](#).

Alat baris perintah secara otomatis menggunakan titik akhir default untuk layanan di AWS Wilayah. Anda dapat menentukan titik akhir yang berbeda untuk permintaan API Anda. Lihat [the section called “Titik akhir Secrets Manager”](#).

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 dan platform pemrograman. SDK mencakup tugas-tugas seperti menandatangani permintaan secara kriptografis, mengelola kesalahan, dan mencoba ulang 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 Wilayah. AWS Anda dapat menentukan titik akhir yang berbeda untuk permintaan API Anda. Lihat [the section called “Titik akhir Secrets Manager”](#).

Untuk dokumentasi SDK, lihat:

- [C++](#)
- [Go](#)
- [Java](#)
- [JavaScript](#)
- [Kotlin](#)
- [.NET](#)
- [PHP](#)
- [Python \(Boto3\)](#)
- [Ruby](#)
- [Karat](#)
- [SAP ABAP](#)
- [Swift](#)

## 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 mengonversi respons menjadi struktur yang secara sintaksis sesuai dengan bahasa Anda.

Untuk melakukan panggilan HTTPS ke Secrets Manager, Anda terhubung ke???.

## AWS Secrets Manager titik akhir

Untuk terhubung secara terprogram ke Secrets Manager, Anda menggunakan endpoint, URL titik masuk untuk layanan. Secrets Manager endpoint adalah endpoint 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, diperlukan beberapa waktu sebelum waktu panggilan habis dan mencoba lagi dengan IPv4. Untuk mengatasi masalah ini, Anda dapat menonaktifkan IPv6 sepenuhnya atau [bermigrasi ke IPv6](#).

Berikut ini adalah endpoint layanan untuk Secrets Manager. Perhatikan bahwa penamaan berbeda dari konvensi [penamaan dual-stack yang khas](#).

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



Nama Wilayah	Wilayah	Titik Akhir	Protokol
AS Barat (California Utara)	us-west-1	secretsmanager.us-west-1.amazonaws.com	HTTPS
		secretsmanager-fips.us-west-1.amazonaws.com	HTTPS
US West (Oregon)	us-west-2	secretsmanager.us-west-2.amazonaws.com	HTTPS
		secretsmanager-fips.us-west-2.amazonaws.com	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 (Melbourne)	ap-southeast-4	secretsmanager.ap-southeast-4.amazonaws.com	HTTPS
Asia Pasifik (Mumbai)	ap-south-1	secretsmanager.ap-south-1.amazonaws.com	HTTPS

Nama Wilayah	Wilayah	Titik Akhir	Protokol
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 Pacific (Tokyo)	ap-northeast-1	secretsmanager.ap-northeast-1.amazonaws.com	HTTPS
Canada (Central)	ca-central-1	secretsmanager.ca-central-1.amazonaws.com	HTTPS
		secretsmanager-fips.ca-central-1.amazonaws.com	HTTPS
Kanada Barat (Calgary)	ca-west-1	secretsmanager.ca-west-1.amazonaws.com	HTTPS
		secretsmanager-fips.ca-west-1.amazonaws.com	HTTPS
Eropa (Frankfurt)	eu-central-1	secretsmanager.eu-central-1.amazonaws.com	HTTPS

Nama Wilayah	Wilayah	Titik Akhir	Protokol
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
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
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

Nama Wilayah	Wilayah	Titik Akhir	Protokol
Amerika Selatan (Sao Paulo)	sa-east-1	secretsmanager.sa-east-1.amazonaws.com	HTTPS
AWS GovCloud (AS-Timur)	us-gov-east-1	secretsmanager.us-gov-east-1.amazonaws.com	HTTPS
		secretsmanager-fips.us-gov-east-1.amazonaws.com	HTTPS
AWS GovCloud (AS-Barat)	us-gov-west-1	secretsmanager.us-gov-west-1.amazonaws.com	HTTPS
		secretsmanager-fips.us-gov-west-1.amazonaws.com	HTTPS

# 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 [Putar rahasia](#).

Secrets Manager menggunakan kebijakan izin IAM untuk memastikan bahwa hanya pengguna yang berwenang yang dapat mengakses atau memodifikasi rahasia. Lihat [Otentikasi dan kontrol akses untuk AWS Secrets Manager](#).

Rahasia memiliki versi yang menyimpan salinan nilai rahasia terenkripsi. Ketika Anda mengubah nilai rahasia, atau rahasia diputar, 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 [Replikasi rahasia di seluruh Wilayah](#).

Lihat [Buat dan kelola rahasia](#).

## Versi rahasia

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

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

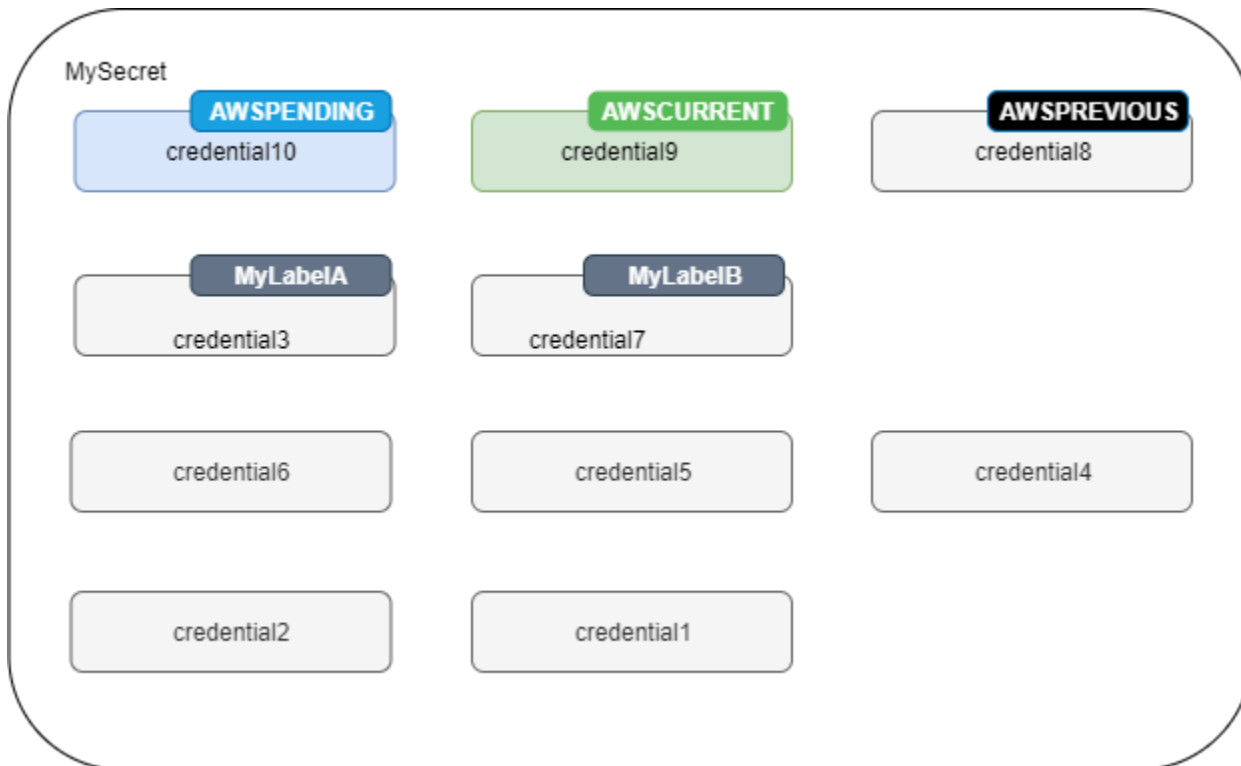
- Versi saat ini - AWSCURRENT
- Versi sebelumnya - AWSPREVIOUS
- Versi yang tertunda (selama rotasi) - AWSPENDING

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

Anda juga dapat memberi label 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 usang. Secrets Manager menghapus versi usang ketika 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.



# Tutorial AWS Secrets Manager

## Topik

- [Temukan rahasia yang tidak dilindungi dalam kode Anda dengan Amazon Reviewer CodeGuru](#)
- [Pindahkan rahasia hardcoded ke AWS Secrets Manager](#)
- [Pindahkan kredensial basis data hardcoded ke AWS Secrets Manager](#)
- [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 hardcoded, ambil tindakan untuk menggantinya:

- [the section called “Ganti kredensial DB hardcoded”](#)
- [the section called “Ganti rahasia hardcoded”](#)

## Pindahkan rahasia hardcoded ke AWS Secrets Manager

Jika Anda memiliki rahasia plaintext dalam kode Anda, kami sarankan Anda memutarkannya 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 hardcoded saat ini sehingga tidak lagi valid.

Untuk rahasia kredensial database, lihat [Pindahkan kredensial basis data hardcoded 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, lihat [the section called “Langkah selanjutnya”](#).

Langkah:

- [Langkah 1: Buat rahasia](#)
- [Langkah 2: Perbarui kode Anda](#)
- [Langkah 3: Perbarui rahasianya](#)
- [Langkah selanjutnya](#)

## Langkah 1: Buat rahasia

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 Store a new secret (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 pasangan kunci/nilai atau di Plaintext. Beberapa contoh:

Pasangan kunci/nilai kunci API:

**ClientID:** *my\_client\_id*

**ClientSecret** : *wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY*

Pasangan kunci/nilai kredensial:

**Username:** *saanvis*

**Password:** *CONTOH-KATA SANDI*

Teks biasa token OAuth:

*AKIAI44QH8DHBEXAMPLE*

Sertifikat digital plaintext:

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

Plaintext kunci pribadi:

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

- c. Untuk kunci Enkripsi, pilih `aws/secretsmanager` untuk menggunakan for Kunci yang dikelola AWS Secrets Manager. 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, lihat [Harga](#).
  - d. Pilih Selanjutnya.
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.

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

- c. Di bagian bawah halaman, pilih Selanjutnya.
5. Pada halaman Konfigurasi rotasi, matikan rotasi. Pilih Selanjutnya.
  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 hardcoded 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 hardcoded.

## Langkah 3: Perbarui rahasianya

Langkah terakhir adalah mencabut dan memperbarui rahasia hardcoded. 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 hardcoded dari kode Anda, beberapa ide untuk dipertimbangkan selanjutnya:

- [Untuk menemukan rahasia hardcoded 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 [Replikasi rahasia di seluruh Wilayah](#).
- 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 “Contoh kebijakan izin”](#)
- Dalam tutorial ini, Anda diberikan izin untuk *RoleToRetrieveSecretAtRuntime* menggunakan kebijakan sumber daya rahasia. Untuk cara lain untuk memberikan izin, lihat [the section called “Melampirkan kebijakan izin ke identitas”](#).

# Pindahkan kredensial basis data hardcoded ke AWS Secrets Manager

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

Untuk database Amazon RDS, Amazon Redshift, dan Amazon DocumentDB, gunakan langkah-langkah di halaman ini untuk memindahkan kredensial hardcoded ke Secrets Manager. Untuk jenis kredensial dan rahasia lainnya, lihat [the section called “Ganti rahasia hardcoded”](#)

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 kredensial saat runtime, *RoleToRetrieveSecretAtRuntime* dalam tutorial ini. Kode Anda mengasumsikan peran ini untuk mengambil rahasia.

Langkah:

- [Langkah 1: Buat Rahasiannya](#)
- [Langkah 2: Perbarui kode Anda](#)
- [Langkah 3: Putar rahasiannya](#)
- [Langkah selanjutnya](#)

## Langkah 1: Buat Rahasiannya

Langkah pertama adalah menyalin kredensial hardcoded 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 Store a new secret (Simpan rahasia baru).
3. Pada halaman Pilih jenis rahasia, lakukan hal berikut:
  - a. Untuk tipe Rahasia, pilih jenis kredensial database yang akan 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 kredensial hardcode yang ada untuk database.
  - c. Untuk kunci Enkripsi, pilih aws/secretsmanager untuk menggunakan for Kunci yang dikelola AWS Secrets Manager. 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, lihat [Harga](#).
  - 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.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::AccountId: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 hardcoded 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 hardcoded.

## Langkah 3: Putar rahasianya

Langkah terakhir adalah mencabut kredensi hardcoded dengan memutar rahasianya. Rotasi adalah proses memperbarui rahasia secara berkala. Ketika Anda memutar rahasia, Anda memperbarui kredensyal di kedua 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 [Titik akhir VPC](#).

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 hardcoded 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 hardcoded di aplikasi Java dan Python Anda, kami merekomendasikan Amazon Reviewer. CodeGuru](#)

## 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 kredensial 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, lihat [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 Generasi otomatis, 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 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 **RDS Tutorial Endpoint**.
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 Auto-assign IP 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 dikelola](#).

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 kredensial 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, aplikasi berbasis GUI. [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 SecretsManagerTutorialKeyPair.pem yang Anda download 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.

```
CREATE DATABASE myDB;  
CREATE USER 'appuser'@'%' IDENTIFIED BY 'EXAMPLE-PASSWORD';  
GRANT ALL PRIVILEGES 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 Store a new secret (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 Selanjutnya.
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 Selanjutnya.

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 kredensial 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 tabel kunci/Nilai, 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. 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.

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 buka [the 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 petunjuk, 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, lihat [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 kredensial 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.

```
CREATE USER 'dbuser'@'%' IDENTIFIED BY 'EXAMPLE-PASSWORD';  
GRANT ALL PRIVILEGES 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 kredensial pengguna yang baru saja Anda buat, dan Anda mengaktifkan rotasi otomatis, termasuk rotasi langsung. Secrets Manager memutar rahasia, yang berarti kata sandi dihasilkan 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 Store a new secret (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 Selanjutnya.
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 Selanjutnya.
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 tersebut masih berisi kredensial yang valid. Kata sandi dalam rahasia telah berubah dari kredensial 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 tabel kunci/Nilai, 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. 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.

## 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, lihat [the 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”](#).

# Otentikasi dan kontrol 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](#).

## 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 hal 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 “Melampirkan kebijakan izin ke identitas”](#)
- [the section called “Lampirkan kebijakan izin ke rahasia”](#)

## 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 [mengkripsi 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”](#).

## Melampirkan kebijakan izin ke identitas

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 kredensi 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.

Untuk informasi selengkapnya, lihat [the section called “Contoh kebijakan izin”](#).

## Lampirkan kebijakan izin ke rahasia AWS Secrets Manager

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 ke beberapa pengguna dan peran.
- Berikan akses ke pengguna atau peran di AWS akun lain.

Lihat [the section called “Contoh kebijakan izin”](#).

Saat Anda melampirkan kebijakan berbasis sumber daya ke rahasia di konsol, Secrets Manager menggunakan mesin penalaran otomatis [Zelkova](#) dan API `ValidateResourcePolicy` untuk mencegah Anda memberikan berbagai kepala sekolah IAM akses 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 secara langsung melekat pada rahasia Anda. Selain menggunakan fitur-fitur ini, periksa dengan cermat kebijakan berikut untuk mengonfirmasi bahwa mereka tidak memberikan akses publik:

- Kebijakan berbasis identitas yang dilampirkan pada AWS prinsipal terkait (misalnya, peran IAM)
- Kebijakan berbasis sumber daya yang dilampirkan pada AWS sumber daya terkait (misalnya, AWS Key Management Service ( ) kunci)AWS KMS

Untuk meninjau izin ke rahasia Anda, lihat [Tentukan 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 Ikhtisar, di bagian Izin sumber daya, pilih Edit izin.
4. Di bidang kode, lakukan salah satu hal berikut, lalu pilih Simpan:
  - Untuk melampirkan atau mengubah kebijakan sumber daya, masukkan kebijakan.
  - 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 pada 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 [Memuat AWS CLI 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`:

```
{  
  "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 pada rahasia, gunakan [GetResourcePolicy](#).

Untuk menghapus kebijakan yang dilampirkan pada rahasia, gunakan [DeleteResourcePolicy](#).

Untuk melampirkan kebijakan ke rahasia, gunakan [PutResourcePolicy](#). Jika sudah ada kebijakan yang dilampirkan, perintah menggantinya dengan kebijakan baru. Kebijakan harus diformat sebagai teks terstruktur JSON. Lihat [Struktur dokumen kebijakan JSON](#). Gunakan [the section called “Contoh kebijakan izin”](#) untuk mulai menulis kebijakan Anda.

Untuk informasi selengkapnya, lihat [the section called “AWS SDK”](#).

## AWS kebijakan terkelola untuk AWS Secrets Manager

Kebijakan AWS terkelola adalah kebijakan mandiri yang dibuat dan dikelola oleh AWS. AWS Kebijakan terkelola dirancang untuk memberikan izin bagi banyak kasus penggunaan umum sehingga Anda dapat mulai menetapkan izin kepada pengguna, grup, dan peran.

Perlu diingat bahwa kebijakan AWS terkelola mungkin tidak memberikan izin hak istimewa paling sedikit untuk kasus penggunaan spesifik Anda karena tersedia untuk digunakan semua pelanggan. AWS Kami menyarankan Anda untuk mengurangi izin lebih lanjut dengan menentukan [kebijakan yang dikelola pelanggan](#) yang khusus untuk kasus penggunaan Anda.

Anda tidak dapat mengubah izin yang ditentukan dalam kebijakan AWS terkelola. Jika AWS memperbarui izin yang ditentukan dalam kebijakan AWS terkelola, pemutakhiran akan memengaruhi semua identitas utama (pengguna, grup, dan peran) yang dilampirkan kebijakan tersebut. AWS kemungkinan besar akan memperbarui kebijakan AWS terkelola saat baru Layanan AWS diluncurkan atau operasi API baru tersedia untuk layanan yang ada.

Untuk informasi selengkapnya, lihat [Kebijakan yang dikelola AWS](#) dalam Panduan Pengguna IAM.

## AWS kebijakan terkelola: SecretsManagerReadWrite

Kebijakan ini menyediakan akses baca/tulis AWS Secrets Manager, termasuk izin untuk menjelaskan sumber daya Amazon RDS, Amazon Redshift, dan Amazon DocumentDB, serta izin yang digunakan untuk mengenkripsi dan mendekripsi rahasia. AWS KMS Kebijakan ini juga memberikan izin untuk membuat set AWS CloudFormation perubahan, mendapatkan templat rotasi dari bucket Amazon S3 yang dikelola oleh AWS, mencantumkan 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 AWS CloudFormation tumpukan dan peran AWS Lambda eksekusi. Anda dapat menetapkan kebijakan FullAccess terkelola [IAM](#). Lihat [Izin untuk rotasi](#).

### Detail izin

Kebijakan ini mencakup izin berikut.

- `secretsmanager`— Memungkinkan kepala sekolah untuk melakukan semua tindakan Secrets Manager.
- `cloudformation`— Memungkinkan kepala sekolah untuk membuat tumpukan. AWS CloudFormation Ini diperlukan agar prinsipal yang menggunakan konsol untuk mengaktifkan rotasi dapat membuat fungsi rotasi Lambda melalui tumpukan. AWS CloudFormation Untuk informasi selengkapnya, lihat [the section called “Bagaimana Secrets Manager menggunakan AWS CloudFormation”](#).

- `ec2`- Memungkinkan kepala sekolah untuk menggambarkan VPC Amazon EC2. 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 kepala sekolah 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 kepala sekolah untuk menggambarkan cluster dan instance di Amazon RDS. Ini diperlukan agar prinsipal yang menggunakan konsol dapat memilih cluster atau instance Amazon RDS.
- `redshift`— Memungkinkan kepala sekolah untuk menggambarkan cluster di Amazon Redshift. Ini diperlukan agar prinsipal yang menggunakan konsol dapat memilih cluster Amazon Redshift.
- `redshift-serverless`— Memungkinkan kepala sekolah untuk mendeskripsikan ruang nama di Amazon Redshift Tanpa Server. Ini diperlukan agar prinsipal yang menggunakan konsol dapat memilih ruang nama Amazon Redshift Tanpa Server.
- `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 diberi tag.
- `serverlessrepo`— Memungkinkan kepala sekolah untuk membuat AWS CloudFormation set perubahan. Ini diperlukan agar prinsipal yang menggunakan konsol dapat membuat fungsi rotasi Lambda. Untuk informasi selengkapnya, lihat [the section called “Bagaimana Secrets Manager menggunakan AWS CloudFormation”](#).
- `s3`— Memungkinkan prinsipal untuk mendapatkan objek dari bucket Amazon S3 yang dikelola oleh AWS Ember ini berisi Lambda [Templat fungsi rotasi](#). Izin ini diperlukan agar prinsipal yang menggunakan konsol dapat membuat fungsi rotasi Lambda berdasarkan templat di bucket. Untuk informasi selengkapnya, lihat [the section called “Bagaimana Secrets Manager menggunakan AWS CloudFormation”](#).

Untuk melihat kebijakan, lihat [dokumen kebijakan SecretsManagerReadWrite JSON](#).

## Secrets Manager memperbarui kebijakan AWS terkelola

Lihat detail tentang pembaruan kebijakan AWS terkelola untuk Secrets Manager.

Perubahan	Deskripsi	Tanggal
<a href="#">SecretsManagerReadWrite</a> — Pembaruan ke kebijakan yang sudah ada	Kebijakan ini diperbarui untuk mengizinkan akses deskripsi ke Amazon Redshift Tanpa Server sehingga pengguna konsol dapat memilih namespace Amazon Redshift Tanpa Server saat mereka membuat rahasia Amazon Redshift.	Maret 12, 2024
<a href="#">SecretsManagerReadWrite</a> – Pembaruan ke kebijakan yang ada	Kebijakan ini diperbarui untuk memungkinkan akses deskripsikan ke kluster elastis Amazon DocumentD B sehingga pengguna konsol dapat memilih kluster elastis saat mereka membuat rahasia Amazon DocumentDB.	12 September 2023
<a href="#">SecretsManagerReadWrite</a> – Pembaruan ke kebijakan yang ada	Kebijakan ini diperbarui untuk mengizinkan akses deskripsikan ke Amazon Redshift sehingga pengguna konsol dapat memilih kluster Amazon Redshift saat mereka membuat rahasia Amazon Redshift. Pembaruan juga menambahkan izin baru untuk memungkinkan akses baca ke bucket Amazon S3 yang dikelola AWS oleh yang	24 Juni 2020

Perubahan	Deskripsi	Tanggal
	menyimpan templat fungsi rotasi Lambda.	
<a href="#">SecretsManagerReadWrite</a> – Pembaruan ke kebijakan yang ada	Kebijakan ini diperbarui untuk mengizinkan akses deskripsi ke kluster Amazon RDS sehingga pengguna konsol dapat memilih kluster saat mereka membuat rahasia Amazon RDS.	3 Mei 2018
<a href="#">SecretsManagerReadWrite</a> – Kebijakan baru	Secrets Manager membuat kebijakan untuk memberikan izin yang diperlukan untuk menggunakan konsol dengan semua akses baca/tulis ke Secrets Manager.	4 April 2018
Secrets Manager mulai melacak perubahan	Secrets Manager mulai melacak perubahan untuk kebijakan yang AWS dikelola.	4 April 2018

## Tentukan siapa yang memiliki izin untuk rahasia Anda AWS Secrets Manager

Secara default, identitas IAM tidak memiliki izin untuk mengakses rahasia. Saat mengotorisasi akses ke rahasia, Secrets Manager mengevaluasi kebijakan berbasis sumber daya yang dilampirkan pada rahasia dan semua kebijakan berbasis identitas yang dilampirkan pada pengguna IAM atau peran 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.

Jika beberapa kebijakan berlaku untuk permintaan, Secrets Manager menggunakan hierarki untuk mengontrol izin:

1. Jika pernyataan dalam kebijakan apa pun dengan eksplisit deny cocok dengan tindakan permintaan dan sumber daya:

Eksplisit deny mengesampingkan semua yang lain dan memblokir tindakan.

2. Jika tidak ada eksplisitdeny, tetapi pernyataan dengan eksplisit allow cocok dengan tindakan permintaan dan sumber daya:

Eksplisit allow memberikan tindakan dalam permintaan akses ke sumber daya dalam pernyataan.

Jika identitas dan rahasia ada dalam dua akun yang berbeda, harus ada kebijakan sumber daya untuk rahasia dan kebijakan yang dilampirkan pada identitas, jika tidak AWS menolak permintaan tersebut. allow Untuk informasi selengkapnya, lihat [Akses lintas akun](#).

3. Jika tidak ada pernyataan dengan eksplisit allow yang cocok dengan tindakan permintaan dan sumber daya:

AWSmenolak permintaan secara default, yang disebut penolakan implisit.

Untuk melihat kebijakan berbasis sumber daya untuk rahasia

- Lakukan salah satu dari 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 to call [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.



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, lihat [the section called “Memodifikasi rahasia”](#).

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 di Account2 untuk mengakses rahasia di Account1.* Untuk menggunakan kebijakan ini, lihat [the section called “Lampirkan kebijakan izin ke rahasia”](#).

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::Account2:role/ApplicationRole"
      },
      "Action": "secretsmanager:GetSecretValue",
      "Resource": "*"
    }
  ]
}
```

Langkah 2: Tambahkan pernyataan ke kebijakan kunci untuk kunci KMS di Akun1

- *Pernyataan kebijakan kunci berikut memungkinkan ApplicationRole di Account2 untuk menggunakan kunci KMS di 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](#).

```
{
  "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 di Account2 untuk mengakses rahasia di Account1 dan mendekripsi nilai rahasia dengan menggunakan kunci enkripsi yang juga ada di Account1.* Untuk menggunakan kebijakan ini, lihat [the section called “Melampirkan kebijakan izin ke identitas”](#). 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](#).

```

{
  "Version" : "2012-10-17",
  "Statement" : [
    {
      "Effect": "Allow",
      "Action": "secretsmanager:GetSecretValue",
      "Resource": "SecretARN"
    },
    {
      "Effect": "Allow",
      "Action": "kms:Decrypt",
      "Resource": "arn:aws:kms:Region:Account1:key/EncryptionKey"
    }
  ]
}

```

## Mengakses rahasia dari lingkungan lokal

Anda dapat menggunakan AWS Identity and Access Management Roles Anywhere untuk mendapatkan kredensial 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 kredensial yang dapat diakses oleh sumber daya di AWS serta perangkat lokal seperti server aplikasi. Untuk informasi selengkapnya, lihat [Panduan Pengguna IAM Roles Anywhere](#).

## Contoh kebijakan izin untuk AWS Secrets Manager

Kebijakan izin adalah teks terstruktur JSON. Lihat [Struktur dokumen kebijakan JSON](#).

Kebijakan izin yang Anda lampirkan ke sumber daya dan identitas sangat mirip. Beberapa elemen yang Anda sertakan dalam kebijakan untuk akses ke rahasia meliputi:

- **Principal:** siapa yang memberikan akses ke. Lihat [Menentukan prinsipal](#) dalam Panduan Pengguna IAM. Ketika Anda melampirkan kebijakan ke identitas, Anda tidak menyertakan **Principal** elemen dalam kebijakan.
- **Action:** apa yang bisa mereka lakukan. Lihat [the section called “Tindakan Secrets Manager”](#).
- **Resource:** rahasia mana yang dapat mereka akses. Lihat [the section called “Sumber daya Secrets Manager”](#).

Karakter wildcard (\*) memiliki arti yang berbeda tergantung pada kebijakan yang Anda lampirkan:

- Dalam kebijakan yang dilampirkan pada rahasia, \* berarti kebijakan tersebut berlaku untuk rahasia ini.
- Dalam kebijakan yang dilampirkan pada identitas, \* berarti kebijakan tersebut berlaku untuk semua sumber daya, termasuk rahasia, di akun.

Untuk melampirkan kebijakan ke rahasia, lihat [the section called “Lampirkan kebijakan izin ke rahasia”](#).

Untuk melampirkan kebijakan ke identitas, lihat [the section called “Melampirkan kebijakan izin ke identitas”](#).

### Topik

- [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 dan VPC](#)
- [Contoh: Kontrol akses ke rahasia menggunakan tag](#)
- [Contoh: Batasi akses ke identitas dengan tag yang cocok dengan tag rahasia](#)
- [Contoh: Prinsipal 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 Kebijakan berbasis [identitas dan kebijakan berbasis sumber daya](#). Untuk informasi tentang cara melampirkan kebijakan, lihat [the section called “Lampirkan kebijakan izin ke rahasia”](#) dan [the section called “Melampirkan kebijakan izin ke identitas”](#).

Contoh berikut menunjukkan dua cara berbeda untuk memberikan akses ke rahasia. Contoh pertama adalah kebijakan berbasis sumber daya yang dapat Anda lampirkan ke rahasia. Contoh ini berguna ketika Anda ingin memberikan akses ke satu rahasia ke beberapa pengguna atau peran. Contoh kedua adalah kebijakan berbasis identitas yang dapat Anda lampirkan ke pengguna atau peran di IAM. Contoh ini berguna ketika Anda ingin memberikan akses ke grup IAM. Untuk memberikan izin untuk mengambil sekelompok rahasia dalam panggilan API batch, lihat [the section called “Contoh: Izin untuk mengambil sekelompok nilai rahasia dalam batch”](#).

Example Baca satu rahasia (lampirkan ke rahasia)

Anda dapat memberikan akses ke rahasia dengan melampirkan kebijakan berikut ke rahasia. Untuk menggunakan kebijakan ini, lihat [the section called “Lampirkan kebijakan izin ke rahasia”](#).

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::AccountId:role/EC2RoleToAccessSecrets"
      },
      "Action": "secretsmanager:GetSecretValue",
      "Resource": "*"
    }
  ]
}
```

```
]
}
```

Example Baca rahasia yang dienkripsi menggunakan kunci yang dikelola pelanggan (lampirkan ke identitas)

Jika rahasia dienkripsi menggunakan kunci yang dikelola pelanggan, Anda dapat memberikan akses untuk membaca rahasia dengan melampirkan kebijakan berikut ke identitas. Untuk menggunakan kebijakan ini, lihat [the section called “Melampirkan kebijakan izin ke identitas”](#).

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "secretsmanager:GetSecretValue",
      "Resource": "SecretARN"
    },
    {
      "Effect": "Allow",
      "Action": "kms:Decrypt",
      "Resource": "KMSKeyARN"
    }
  ]
}
```

## Contoh: Izin untuk membaca dan menggambarkan rahasia individu

Example Baca dan jelaskan satu rahasia (lampirkan pada identitas)

Anda dapat memberikan akses ke rahasia dengan melampirkan kebijakan berikut ke identitas. Untuk menggunakan kebijakan ini, lihat [the section called “Melampirkan kebijakan izin ke identitas”](#).

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:GetSecretValue",
        "secretsmanager:DescribeSecret"
      ],

```

```

    "Resource": "SecretARN"
  }
]
}

```

## Contoh: Izin untuk mengambil sekelompok nilai rahasia dalam batch

Example Baca sekelompok rahasia dalam batch (lampirkan ke identitas)

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*, *SecretArn2*, dan *SecretArn3*, 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](#). Untuk menggunakan kebijakan ini, lihat [the section called "Melampirkan kebijakan izin ke identitas"](#).

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:BatchGetSecretValue",
        "secretsmanager:ListSecrets"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:GetSecretValue"
      ],
      "Resource": [
        "SecretARN1",
        "SecretARN2",
        "SecretARN3"
      ]
    }
  ]
}

```

## Contoh: Wildcard

Anda dapat menggunakan wildcard untuk menyertakan sekumpulan nilai dalam elemen kebijakan.

Example Akses semua rahasia di jalur (lampirkan ke identitas)

Kebijakan berikut memberikan akses untuk mengambil semua rahasia dengan nama yang diawali dengan *"TestEnv/"*. Untuk menggunakan kebijakan ini, lihat [the section called "Melampirkan kebijakan izin ke identitas"](#).

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "secretsmanager:GetSecretValue",
    "Resource": "arn:aws:secretsmanager:Region:AccountId:secret:TestEnv/*"
  }
}
```

Example Akses metadata pada semua rahasia (lampirkan ke identitas)

Pemberian DescribeSecret dan izin kebijakan berikut dimulai dengan List: ListSecrets dan ListSecretVersionIds Untuk menggunakan kebijakan ini, lihat [the section called "Melampirkan kebijakan izin ke identitas"](#).

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "secretsmanager:DescribeSecret",
      "secretsmanager:List*"
    ],
    "Resource": "*"
  }
}
```

Example Cocokkan nama rahasia (lampirkan ke identitas)

Kebijakan berikut memberikan semua izin Secrets Manager untuk rahasia berdasarkan nama. Untuk menggunakan kebijakan ini, lihat [the section called "Melampirkan kebijakan izin ke identitas"](#).

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.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "secretsmanager:*",
      "Resource": [
        "arn:aws:secretsmanager:Region:AccountId:secret:a_specific_secret_name-a1b2c3",
        "arn:aws:secretsmanager:Region:AccountId: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 (lampirkan ke identitas)

Kebijakan berikut memberikan izin untuk membuat rahasia dan melihat daftar rahasia. Untuk menggunakan kebijakan ini, lihat [the section called “Melampirkan kebijakan izin ke identitas”](#).

```
{
  "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** (lampirkan ke identitas)

Kebijakan berikut menunjukkan cara menolak penggunaan kunci AWS terkelola `aws/secretsmanager` untuk membuat atau memperbarui rahasia. Ini berarti bahwa rahasia harus dienkripsi menggunakan kunci yang dikelola pelanggan. Jika `aws/secretsmanager` kuncinya ada, Anda juga harus menyertakan ID kuncinya. Anda juga menyertakan string kosong karena Secrets Manager menafsirkannya sebagai kunci AWS `aws/secretsmanager` terkelola. Pernyataan kedua menolak permintaan untuk membuat rahasia yang tidak menyertakan kunci KMS, karena Secrets Manager menafsirkannya sebagai kunci AWS terkelola. `aws/secretsmanager`

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "RequireCustomerManagedKeysOnSecrets",
      "Effect": "Deny",
      "Action": [
        "secretsmanager:CreateSecret",
        "secretsmanager:UpdateSecret"
      ]
    }
  ]
}

```

```

    ],
    "Resource": "*",
    "Condition": {
      "ForAnyValue:StringLikeIfExists": {
        "secretsmanager:KmsKeyId": [
          "*alias/aws/secretsmanager",
          "<key_ID_of_the_AWS_managed_key>",
          ""
        ]
      }
    }
  },
  {
    "Sid": "RequireKmsKeyIdParameterOnCreate",
    "Effect": "Deny",
    "Action": "secretsmanager:CreateSecret",
    "Resource": "*",
    "Condition": {
      "Null": {
        "secretsmanager:KmsKeyId": "true"
      }
    }
  }
]
}

```

## Contoh: Izin dan VPC

Jika Anda perlu mengakses Secrets Manager dari dalam VPC, Anda dapat memastikan bahwa permintaan ke Secrets Manager berasal dari VPC dengan menyertakan kondisi dalam kebijakan izin Anda. Untuk informasi selengkapnya, lihat [Kondisi titik akhir VPC](#) dan [Titik akhir VPC](#).

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 (lampirkan ke rahasia)

Kebijakan berikut memungkinkan pengguna untuk melakukan operasi Secrets Manager hanya ketika permintaan datang melalui titik akhir VPC. *vpce-1234a5678b9012c* Untuk menggunakan kebijakan ini, lihat [the section called "Lampirkan kebijakan izin ke rahasia"](#).

```
{
```

```

{
  "Id": "example-policy-1",
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "RestrictGetSecretValueoperation",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "secretsmanager:GetSecretValue",
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
          "aws:sourceVpce": "vpce-1234a5678b9012c"
        }
      }
    }
  ]
}

```

Example Memerlukan permintaan untuk datang dari VPC (lampirkan ke rahasia)

Kebijakan berikut memungkinkan perintah untuk membuat dan mengelola rahasia hanya ketika mereka berasal *vpce-12345678*. Selain itu, kebijakan memungkinkan operasi yang menggunakan akses nilai terenkripsi rahasia hanya ketika permintaan berasal. *vpce-2b2b2b2b* Anda mungkin menggunakan kebijakan seperti ini jika Anda menjalankan aplikasi dalam satu VPC, tetapi Anda menggunakan VPC kedua yang terisolasi untuk fungsi manajemen. Untuk menggunakan kebijakan ini, lihat [the section called "Lampirkan kebijakan izin ke rahasia"](#).

```

{
  "Id": "example-policy-2",
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowAdministrativeActionsfromONLYvpce-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: Kontrol akses ke rahasia menggunakan tag

Anda dapat menggunakan tag untuk mengontrol akses ke rahasia Anda. Menggunakan tag untuk mengontrol izin sangat membantu di lingkungan yang berkembang pesat dan membantu situasi di mana manajemen kebijakan menjadi rumit. Salah satu strategi adalah melampirkan tag ke rahasia dan kemudian memberikan izin ke identitas ketika rahasia memiliki tag tertentu.

Example Izinkan akses ke rahasia dengan tag tertentu (lampirkan ke identitas)

Kebijakan berikut memungkinkan DescribeSecret pada rahasia dengan tag dengan kunci "" dan nilai *ServerName* "ServeABC". Untuk menggunakan kebijakan ini, lihat [the section called "Melampirkan kebijakan izin ke identitas"](#).

```

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

```

```

"Statement": {
  "Effect": "Allow",
  "Action": "secretsmanager:DescribeSecret",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "secretsmanager:ResourceTag/ServerName": "ServerABC"
    }
  }
}
}

```

## Contoh: Batasi akses ke identitas dengan tag yang cocok dengan tag rahasia

Salah satu strateginya adalah melampirkan tag ke rahasia dan identitas IAM. Kemudian Anda membuat kebijakan izin untuk mengizinkan operasi ketika tag identitas cocok dengan tag rahasia. Untuk tutorial selengkapnya, lihat [Menentukan izin untuk mengakses rahasia berdasarkan tag](#).

Menggunakan tag untuk mengontrol izin sangat membantu di lingkungan yang berkembang pesat dan membantu situasi di mana manajemen kebijakan menjadi rumit. Untuk informasi lebih lanjut, lihat [Apa fungsi ABAC untuk AWS?](#)

Example Izinkan akses ke peran yang memiliki tag yang sama dengan rahasia (lampirkan ke rahasia)

Kebijakan berikut `123456789012` hanya `GetSecretValue` memberikan akun jika tag `AccessProject` memiliki nilai yang sama untuk rahasia dan peran. Untuk menggunakan kebijakan ini, lihat [the section called "Lampirkan kebijakan izin ke rahasia"](#).

```

{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {
      "AWS": "123456789012"
    },
    "Condition": {
      "StringEquals": {
        "aws:ResourceTag/AccessProject": "${ aws:PrincipalTag/AccessProject }"
      }
    }
  },
}

```

```
"Action": "secretsmanager:GetSecretValue",
"Resource": "*"
}
}
```

## Contoh: Prinsipal layanan

Jika kebijakan sumber daya yang dilampirkan pada rahasia Anda menyertakan [prinsip AWS layanan](#), kami sarankan Anda menggunakan kunci kondisi SourceAccount global [aws: SourceArn](#) dan [aws: ARN](#) dan nilai akun disertakan dalam konteks otorisasi hanya ketika permintaan datang ke Secrets Manager dari layanan lain. AWS Kombinasi kondisi ini menghindari [skenario wakil yang berpotensi 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 kondisi. `aws: SourceArn` Sebagai gantinya, gunakan tombol `aws: SourceAccount` kondisi. Untuk informasi selengkapnya, lihat [persyaratan IAM](#).

Prinsipal layanan biasanya tidak digunakan sebagai prinsipal dalam kebijakan yang melekat pada rahasia, tetapi beberapa layanan memerlukannya. AWS Untuk informasi tentang kebijakan sumber daya yang diharuskan layanan untuk Anda lampirkan ke rahasia, lihat dokumentasi layanan.

Example Izinkan layanan mengakses rahasia menggunakan kepala layanan (lampirkan ke rahasia)

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "service-name.amazonaws.com"
        ]
      },
      "Action": "secretsmanager:GetSecretValue",
      "Resource": "*",
      "Condition": {
        "ArnLike": {
          "aws:sourceArn": "arn:aws:service-name::123456789012:*"
        }
      },
    }
  ]
}
```

```
    "StringEquals": {
      "aws:sourceAccount": "123456789012"
    }
  }
}
]
```

## Referensi izin untuk AWS Secrets Manager

Untuk melihat elemen yang membentuk kebijakan izin, lihat [Struktur dokumen kebijakan JSON dan referensi elemen kebijakan IAM JSON](#).

Untuk mulai menulis kebijakan izin Anda sendiri, lihat [the section called "Contoh kebijakan izin"](#).

Kolom tipe sumber daya pada tabel Tindakan menunjukkan apakah setiap tindakan mendukung izin tingkat sumber daya. Jika tidak ada nilai untuk kolom ini, Anda harus menentukan semua sumber daya ("\*") yang berlaku kebijakan dalam Resource elemen pernyataan kebijakan Anda. Jika kolom mencantumkan jenis sumber daya, maka Anda dapat menyebutkan ARN dengan jenis tersebut dalam sebuah pernyataan dengan tindakan tersebut. Jika tindakan memiliki satu atau lebih sumber daya yang diperlukan, pemanggil harus memiliki izin untuk menggunakan tindakan dengan sumber daya tersebut. Sumber daya yang diperlukan ditunjukkan dalam tabel dengan tanda bintang (\*). Jika Anda membatasi akses sumber daya dengan Resource elemen dalam kebijakan IAM, Anda harus menyertakan ARN atau pola untuk setiap jenis sumber daya yang diperlukan. Beberapa tindakan mendukung berbagai jenis sumber daya. Jika jenis sumber daya opsional (tidak ditunjukkan sesuai kebutuhan), maka Anda dapat memilih untuk menggunakan salah satu jenis sumber daya opsional.

Kolom Condition keys pada tabel Actions menyertakan kunci yang dapat Anda tentukan dalam Condition elemen pernyataan kebijakan. Untuk informasi selengkapnya tentang kunci kondisi yang terkait dengan sumber daya untuk layanan, lihat kolom Kunci kondisi pada tabel Jenis sumber daya.

## Tindakan Secrets Manager

Tindakan	Deskripsi	Tingkat akses	Jenis sumber daya (*diperlukan)	Kunci syarat	Tindakan bergantung
<a href="#">BatchGetSecretValue</a>	Memberikan izin untuk mengambil dan mendekripsi daftar rahasia	Daftar			
<a href="#">CancelRotateSecret</a>	Memberikan izin untuk membatalkan rotasi rahasia yang sedang berlangsung	Tulis	<a href="#">Secret*</a>	<a href="#">secretsmanager:SecretId</a> <a href="#">secretsmanager:resource/AllowRotationLambdaAction</a> <a href="#">secretsmanager:ResourceTag/tag-key</a> <a href="#">aws:ResourceTag/\${TagKey}</a> <a href="#">secretsmanager:SecretPrimaryRegion</a>	



Tindakan	Deskripsi	Tingkat akses	Jenis sumber daya (*diperlukan)	Kunci syarat	Tindakan bergantung
<a href="#">CreateSecret</a>	Memberikan izin untuk membuat rahasia yang menyimpan data terenkripsi yang dapat ditanyakan dan diputar	Tulis	<a href="#">Secret*</a>	<a href="#">secretsmanager:Name</a> <a href="#">secretsmanager:Description</a> <a href="#">secretsmanager:KeyId</a> <a href="#">aws:RequestTag/\${TagKey}</a> <a href="#">aws:ResourceTag/\${TagKey}</a> <a href="#">aws:TagKeys</a> <a href="#">secretsmanager:ResourceTag/tag-key</a> <a href="#">secretsmanager:Add</a>	

Tindakan	Deskripsi	Tingkat akses	Jenis sumber daya (*diperlukan)	Kunci syarat	Tindakan bergantung
				<a href="#">ReplicaRegions</a> <a href="#">secretsmanager:ForceOverwriteReplicaSecret</a>	
<a href="#">DeleteResourcePolicy</a>	Memberikan izin untuk menghapus kebijakan sumber daya yang dilampirkan pada rahasia	Manajemen izin	<a href="#">Secret*</a>		

Tindakan	Deskripsi	Tingkat akses	Jenis sumber daya (*diperlukan)	Kunci syarat	Tindakan bergantung
				<a href="#">secretsmanager:SecretId</a>  <a href="#">secretsmanager:resource/AllowRotationLambdaArn</a>  <a href="#">secretsmanager:ResourceTag/tag-key</a>  <a href="#">aws:ResourceTag/\${TagKey}</a>  <a href="#">secretsmanager:SecretPrimaryRegion</a>	
<a href="#">DeleteSecret</a>	Memberikan izin untuk menghapus rahasia	Tulis	<a href="#">Secret*</a>		

Tindakan	Deskripsi	Tingkat akses	Jenis sumber daya (*diperlukan)	Kunci syarat	Tindakan bergantung
				<a href="#"><u>secretsmanager:SecretId</u></a>  <a href="#"><u>secretsmanager:resource/AllowRotationLambdaAction</u></a>  <a href="#"><u>secretsmanager:RecoveryWindowInDays</u></a>  <a href="#"><u>secretsmanager:ForceDeleteWithoutRecovery</u></a>  <a href="#"><u>secretsmanager:ResourceTag/tag-key</u></a>  <a href="#"><u>aws:ResourceTag/\${TagKey}</u></a>	

Tindakan	Deskripsi	Tingkat akses	Jenis sumber daya (*diperlukan)	Kunci syarat	Tindakan bergantung
				<a href="#">secretsmanager:SecretPrimaryRegion</a>	
<a href="#">DescribeSecret</a>	Memberikan izin untuk mengambil metadata tentang rahasia, tetapi bukan data terenkripsi	Baca	<a href="#">Secret*</a>	<a href="#">secretsmanager:SecretId</a> <a href="#">secretsmanager:resource/AllowRotationLambdaAction</a> <a href="#">secretsmanager:ResourceTag/tag-key</a> <a href="#">aws:ResourceTag/\${TagKey}</a> <a href="#">secretsmanager:SecretPrimaryRegion</a>	

Tindakan	Deskripsi	Tingkat akses	Jenis sumber daya (*diperlukan)	Kunci syarat	Tindakan bergantung
<a href="#">GetRandomPassword</a>	Memberikan izin untuk menghasilkan string acak untuk digunakan dalam pembuatan kata sandi	Baca			
<a href="#">GetResourcePolicy</a>	Memberikan izin untuk mendapatkan kebijakan sumber daya yang melekat pada rahasia	Baca	<a href="#">Secret*</a>	<a href="#">secretsmanager:SecretId</a> <a href="#">secretsmanager:resource/AllowRotationLambdaAction</a> <a href="#">secretsmanager:ResourceTag/tag-key</a> <a href="#">aws:ResourceTag/\${TagKey}</a> <a href="#">secretsmanager:SecretPrimaryRegion</a>	

Tindakan	Deskripsi	Tingkat akses	Jenis sumber daya (*diperlukan)	Kunci syarat	Tindakan bergantung
<a href="#">GetSecretValue</a>	Memberikan izin untuk mengambil dan mendekripsi data terenkripsi	Baca	<a href="#">Secret*</a>		

Tindakan	Deskripsi	Tingkat akses	Jenis sumber daya (*diperlukan)	Kunci syarat	Tindakan bergantung
				<a href="#">secretsmanager:SecretId</a> <a href="#">secretsmanager:VersionId</a> <a href="#">secretsmanager:VersionStage</a> <a href="#">secretsmanager:resource/AllowRotationLambdaAction</a> <a href="#">secretsmanager:ResourceTag/tag-key</a> <a href="#">aws:ResourceTag/\${TagKey}</a> <a href="#">secretsmanager:SecretPrimaryRegion</a>	



Tindakan	Deskripsi	Tingkat akses	Jenis sumber daya (*diperlukan)	Kunci syarat	Tindakan bergantung
<a href="#">ListSecretVersionIds</a>	Memberikan izin untuk membuat daftar versi rahasia yang tersedia	Baca	<a href="#">Secret*</a>	<a href="#">secretsmanager:SecretId</a> <a href="#">secretsmanager:resource/AllowRotationLambdaAction</a> <a href="#">secretsmanager:ResourceTag/tag-key</a> <a href="#">aws:ResourceTag/\${TagKey}</a> <a href="#">secretsmanager:SecretPrimaryRegion</a>	
<a href="#">ListSecrets</a>	Memberikan izin untuk membuat daftar rahasia yang tersedia	Daftar			
<a href="#">PutResourcePolicy</a>		Manajemen izin	<a href="#">Secret*</a>		

Tindakan	Deskripsi	Tingkat akses	Jenis sumber daya (*diperlukan)	Kunci syarat	Tindakan bergantung
	<p>Memberikan izin untuk melampirkan kebijakan sumber daya ke rahasia</p>			<p><a href="#">secretsmanager:SecretId</a></p> <p><a href="#">secretsmanager:resource/AllowRotationLambdaArn</a></p> <p><a href="#">secretsmanager:ResourceTag/tag-key</a></p> <p><a href="#">aws:ResourceTag/\${TagKey}</a></p> <p><a href="#">secretsmanager:BlockPublicPolicy</a></p> <p><a href="#">secretsmanager:SecretPrimaryRegion</a></p>	
<p><a href="#">PutSecretValue</a></p>		<p>Tulis</p>	<p><a href="#">Secret*</a></p>		

Tindakan	Deskripsi	Tingkat akses	Jenis sumber daya (*diperlukan)	Kunci syarat	Tindakan bergantung
	Memberikan izin untuk membuat versi baru rahasia dengan data terenkripsi baru			<a href="#">secretsmanager:SecretId</a>  <a href="#">secretsmanager:resource/AllowRotationLambdaAction</a>  <a href="#">secretsmanager:ResourceTag/tag-key</a>  <a href="#">aws:ResourceTag/\${TagKey}</a>  <a href="#">secretsmanager:SecretPrimaryRegion</a>	
<a href="#">RemoveRegionsFromReplication</a>	Memberikan izin untuk menghapus Wilayah dari replikasi	Tulis	<a href="#">Secret*</a>		

Tindakan	Deskripsi	Tingkat akses	Jenis sumber daya (*diperlukan)	Kunci syarat	Tindakan bergantung
				<a href="#">secretsmanager:SecretId</a>  <a href="#">secretsmanager:resource/AllowRotationLambdaArn</a>  <a href="#">secretsmanager:ResourceTag/tag-key</a>  <a href="#">aws:ResourceTag/\${TagKey}</a>  <a href="#">secretsmanager:SecretPrimaryRegion</a>	

Tindakan	Deskripsi	Tingkat akses	Jenis sumber daya (*diperlukan)	Kunci syarat	Tindakan bergantung
<a href="#">Replicate SecretToRegions</a>	Memberikan izin untuk mengubah rahasia yang ada menjadi rahasia Multi-wilayah dan mulai mereplikasi rahasia ke daftar Wilayah baru	Tulis	<a href="#">Secret*</a>	<a href="#">secretsmanager:SecretId</a>  <a href="#">secretsmanager:resource/AllowRotationLambdaArn</a>  <a href="#">secretsmanager:ResourceTag/tag-key</a>  <a href="#">aws:ResourceTag/\${TagKey}</a>  <a href="#">secretsmanager:SecretPrimaryRegion</a>  <a href="#">secretsmanager:AddReplicaRegions</a>	

Tindakan	Deskripsi	Tingkat akses	Jenis sumber daya (*diperlukan)	Kunci syarat	Tindakan bergantung
				<a href="#">secretsmanager:ForceOverwriteReplicaSecret</a>	
<a href="#">RestoreSecret</a>	Memberikan izin untuk membatalkan penghapusan rahasia	Tulis	<a href="#">Secret*</a>	<a href="#">secretsmanager:SecretId</a> <a href="#">secretsmanager:resource/AllowRotationLambdaArn</a> <a href="#">secretsmanager:ResourceTag/tag-key</a> <a href="#">aws:ResourceTag/\${TagKey}</a> <a href="#">secretsmanager:SecretPrimaryRegion</a>	

Tindakan	Deskripsi	Tingkat akses	Jenis sumber daya (*diperlukan)	Kunci syarat	Tindakan bergantung
<a href="#">RotateSecret</a>	Memberikan izin untuk memulai rotasi rahasia	Tulis	<a href="#">Secret*</a>		

Tindakan	Deskripsi	Tingkat akses	Jenis sumber daya (*diperlukan)	Kunci syarat	Tindakan bergantung
				<a href="#">secretsmanager:SecretId</a> <a href="#">secretsmanager:RotationLambdaARN</a> <a href="#">secretsmanager:resource/AllowRotationLambdaArn</a> <a href="#">secretsmanager:ResourceTag/tag-key</a> <a href="#">aws:ResourceTag/\${TagKey}</a> <a href="#">secretsmanager:SecretPrimaryRegion</a> <a href="#">secretsmanager:Mod</a>	



Tindakan	Deskripsi	Tingkat akses	Jenis sumber daya (*diperlukan)	Kunci syarat	Tindakan bergantung
				<a href="#">ifyRotati onRules</a>  <a href="#">secretsma nager:Rot ateImmedi ately</a>	

Tindakan	Deskripsi	Tingkat akses	Jenis sumber daya (*diperlukan)	Kunci syarat	Tindakan bergantung
<a href="#">StopReplicationToReplica</a>	Memberikan izin untuk menghapus rahasia dari replikasi dan mempromosikan rahasia ke rahasia regional di wilayah replika	Tulis	<a href="#">Secret*</a>	<a href="#">secretsmanager:SecretId</a> <a href="#">secretsmanager:resource/AllowRotationLambdaAction</a> <a href="#">secretsmanager:ResourceTag/tag-key</a> <a href="#">aws:ResourceTag/\${TagKey}</a> <a href="#">secretsmanager:SecretPrimaryRegion</a>	
<a href="#">TagResource</a>	Memberikan izin untuk menambahkan tag ke rahasia	Penandaan	<a href="#">Secret*</a>		

Tindakan	Deskripsi	Tingkat akses	Jenis sumber daya (*diperlukan)	Kunci syarat	Tindakan bergantung
				<a href="#">secretsmanager:SecretId</a> <a href="#">aws:RequestTag/\${TagKey}</a> <a href="#">aws:TagKeys</a> <a href="#">secretsmanager:resource/AllowRotationLambdaAction</a> <a href="#">secretsmanager:ResourceTag/tag-key</a> <a href="#">aws:ResourceTag/\${TagKey}</a> <a href="#">secretsmanager:SecretPrimaryRegion</a>	

Tindakan	Deskripsi	Tingkat akses	Jenis sumber daya (*diperlukan)	Kunci syarat	Tindakan bergantung
<a href="#">UntagResource</a>	Memberikan izin untuk menghapus tag dari rahasia	Penandaan	<a href="#">Secret*</a>	<a href="#">secretsmanager:SecretId</a> <a href="#">aws:TagKeys</a> <a href="#">secretsmanager:resource/AllowRotationLambdaAction</a> <a href="#">secretsmanager:ResourceTag/tag-key</a> <a href="#">aws:ResourceTag/\${TagKey}</a> <a href="#">secretsmanager:SecretPrimaryRegion</a>	

Tindakan	Deskripsi	Tingkat akses	Jenis sumber daya (*diperlukan)	Kunci syarat	Tindakan bergantung
<a href="#">UpdateSecret</a>	Memberikan izin untuk memperbarui rahasia dengan metadata baru atau dengan versi baru dari data terenkripsi	Tulis	<a href="#">Secret*</a>		

Tindakan	Deskripsi	Tingkat akses	Jenis sumber daya (*diperlukan)	Kunci syarat	Tindakan bergantung
				<a href="#">secretsmanager:SecretId</a> <a href="#">secretsmanager:Description</a> <a href="#">secretsmanager:KmsKeyId</a> <a href="#">secretsmanager:resource/AllowRotationLambdaArn</a> <a href="#">secretsmanager:ResourceTag/tag-key</a> <a href="#">aws:ResourceTag/\${TagKey}</a> <a href="#">secretsmanager:SecretPrimaryRegion</a>	

Tindakan	Deskripsi	Tingkat akses	Jenis sumber daya (*diperlukan)	Kunci syarat	Tindakan bergantung
<a href="#">UpdateSecretVersionStage</a>	Memberikan izin untuk memindahkan tanggung dari satu rahasia ke rahasia lainnya	Tulis	<a href="#">Secret*</a>	<a href="#">secretsmanager:SecretId</a> <a href="#">secretsmanager:VersionStage</a> <a href="#">secretsmanager:resource/AllowRotationLambdaAction</a> <a href="#">secretsmanager:ResourceTag/tag-key</a> <a href="#">aws:ResourceTag/\${TagKey}</a> <a href="#">secretsmanager:SecretPrimaryRegion</a>	

Tindakan	Deskripsi	Tingkat akses	Jenis sumber daya (*diperlukan)	Kunci syarat	Tindakan bergantung
<a href="#">ValidateResourcePolicy</a>	Memberikan izin untuk memvalidasi kebijakan sumber daya sebelum melampirkan kebijakan	Manajemen izin	<a href="#">Secret*</a>	<a href="#">secretsmanager:SecretId</a> <a href="#">secretsmanager:resource/AllowRotationLambdaArn</a> <a href="#">secretsmanager:ResourceTag/tag-key</a> <a href="#">aws:ResourceTag/\${TagKey}</a> <a href="#">secretsmanager:SecretPrimaryRegion</a>	



## Sumber daya Secrets Manager

Jenis sumber daya	ARN	Kunci syarat
<a href="#">Secret</a>	<code>arn:\${Partition}:secretsmanager:\${Region}:\${Account}:secret:\${SecretId}</code>	<a href="#">aws:RequestTag/\${TagKey}</a>  <a href="#">aws:ResourceTag/\${TagKey}</a>  <a href="#">aws:TagKeys</a>  <a href="#">secretsmanager:ResourceTag/tag-key</a>  <a href="#">secretsmanager:resource/AllowRotationLambdaArn</a>

Secrets Manager membangun bagian terakhir dari ARN rahasia dengan menambahkan tanda hubung dan enam karakter alfanumerik acak di akhir nama rahasia. Jika Anda menghapus rahasia dan kemudian membuat ulang rahasia lain dengan nama yang sama, pemformatan ini membantu memastikan bahwa individu dengan izin ke rahasia asli tidak secara otomatis mendapatkan akses ke rahasia baru karena Secrets Manager menghasilkan enam karakter acak baru.

Anda dapat menemukan ARN untuk rahasia di konsol Secrets Manager di halaman detail rahasia atau dengan menelepon. [DescribeSecret](#)

### Kunci syarat

Jika Anda menyertakan kondisi string dari tabel berikut dalam kebijakan izin, penelepon ke Secrets Manager harus meneruskan parameter yang cocok atau mereka ditolak aksesnya. Untuk menghindari penolakan penelepon untuk parameter yang hilang, tambahkan `IfExists` ke akhir nama operator kondisi, misalnya. `StringLikeIfExists` Untuk informasi selengkapnya, lihat [elemen kebijakan IAM JSON: Operator kondisi](#).

Kunci syarat	Deskripsi	Jenis
<a href="#">aws:RequestTag/\${TagKey}</a>	Memfilter akses dengan kunci yang ada dalam permintaan yang dibuat pengguna ke layanan Secrets Manager	String
<a href="#">aws:ResourceTag/\${TagKey}</a>	Filter akses dengan tanda yang terkait dengan sumber daya	String
<a href="#">aws:TagKeys</a>	Memfilter akses berdasarkan daftar semua nama kunci tag yang ada dalam permintaan yang dibuat pengguna ke layanan Secrets Manager	ArrayOfString
<a href="#">secretsmanager:AddReplicaRegions</a>	Memfilter akses berdasarkan daftar Wilayah untuk mereplikasi rahasia	ArrayOfString
<a href="#">secretsmanager:BlockPublicPolicy</a>	Memfilter akses berdasarkan apakah kebijakan sumber daya memblokir Akun AWS akses luas	Bool
<a href="#">secretsmanager:Description</a>	Memfilter akses dengan teks deskripsi dalam permintaan	String
<a href="#">secretsmanager:ForceDeleteWithoutRecovery</a>	Memfilter akses dengan apakah rahasianya akan segera dihapus tanpa jendela pemulihan	Bool
<a href="#">secretsmanager:ForceOverwriteReplicaSecret</a>	Memfilter akses berdasarkan apakah akan menimpa rahasia dengan nama yang sama di Wilayah tujuan	Bool
<a href="#">secretsmanager:KmsKeyId</a>	Memfilter akses oleh ARN dari kunci KMS dalam permintaan	String

Kunci syarat	Deskripsi	Jenis
<a href="#">secretsmanager:ModifyRotationRules</a>	Memfilter akses dengan apakah aturan rotasi rahasia akan dimodifikasi	Bool
<a href="#">secretsmanager:Name</a>	Memfilter akses dengan nama rahasia yang ramah dalam permintaan	String
<a href="#">secretsmanager:RecoveryWindowInDays</a>	Memfilter akses berdasarkan jumlah hari yang menunggu Secrets Manager sebelum dapat menghapus rahasia	Numerik
<a href="#">secretsmanager:ResourceTag/tag-key</a>	Memfilter akses dengan kunci tag dan pasangan nilai	String
<a href="#">secretsmanager:RotateImmediately</a>	Memfilter akses dengan apakah rahasianya akan segera diputar	Bool
<a href="#">secretsmanager:RotationLambdaARN</a>	Memfilter akses oleh ARN dari fungsi Lambda rotasi dalam permintaan	ARN
<a href="#">secretsmanager:SecretId</a>	Memfilter akses berdasarkan nilai secretID dalam permintaan	ARN
<a href="#">secretsmanager:SecretPrimaryRegion</a>	Memfilter akses oleh Wilayah utama tempat rahasia dibuat	String
<a href="#">secretsmanager:VersionId</a>	Memfilter akses oleh pengenal unik dari versi rahasia dalam permintaan	String

Kunci syarat	Deskripsi	Jenis
<a href="#">secretsmanager:VersionStage</a>	Memfilter akses berdasarkan daftar tahapan versi dalam permintaan	String
<a href="#">secretsmanager:resource/AllowRotationLambdaArn</a>	Memfilter akses oleh ARN dari fungsi rotasi Lambda yang terkait dengan rahasia	ARN

## Blokir akses luas ke rahasia dengan **BlockPublicPolicy** kondisi

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`.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "secretsmanager:PutResourcePolicy",
    "Resource": "SecretId",
    "Condition": {
      "Bool": {
        "secretsmanager:BlockPublicPolicy": "true"
      }
    }
  }
}
```

## Kondisi alamat IP

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 “Kondisi titik akhir VPC”](#).

## 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 [Titik akhir VPC](#).

# Buat dan kelola rahasia dengan AWS Secrets Manager

Rahasia dapat berupa kata sandi, seperangkat kredensial seperti nama pengguna dan kata sandi, token OAuth, atau informasi rahasia lainnya yang Anda simpan dalam bentuk terenkripsi di Secrets Manager.

## Topik

- [Buat rahasia AWS Secrets Manager database](#)
- [Struktur rahasia JSON AWS Secrets Manager](#)
- [Buat AWS Secrets Manager rahasia](#)
- [Perbarui nilai untuk AWS Secrets Manager rahasia](#)
- [Buat kata sandi dengan Secrets Manager](#)
- [Kembalikan rahasia ke versi sebelumnya](#)
- [Ubah kunci enkripsi untuk AWS Secrets Manager rahasia](#)
- [Memodifikasi AWS Secrets Manager rahasia](#)
- [Temukan rahasia di AWS Secrets Manager](#)
- [Hapus AWS Secrets Manager rahasia](#)
- [Kembalikan AWS Secrets Manager rahasia](#)
- [Tag AWS Secrets Manager rahasia](#)

## Buat rahasia AWS Secrets Manager database

Setelah Anda membuat pengguna di Amazon RDS, Amazon Aurora, Amazon Redshift, atau Amazon DocumentDB, Anda dapat menyimpan kredensialnya di Secrets Manager dengan mengikuti langkah-langkah ini. Ketika Anda menggunakan AWS CLI atau salah satu SDK untuk menyimpan rahasia, Anda harus memberikan rahasia dalam struktur [JSON yang benar](#). Saat Anda menggunakan konsol untuk menyimpan rahasia database, Secrets Manager secara otomatis membuatnya dalam struktur JSON yang benar.

**Tip**

Untuk kredensial pengguna admin Amazon RDS dan Amazon Redshift, kami sarankan [Anda menggunakan rahasia terkelola](#). Anda membuat rahasia terkelola melalui service pengelolaan, dan kemudian Anda dapat menggunakan [rotasi terkelola](#).

Ketika Anda menyimpan kredensial 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 pasangan kunci/nilai tambahan ke rahasia untuk informasi koneksi regional.

Untuk membuat rahasia, Anda memerlukan izin yang diberikan oleh [SecretsManagerReadWrite](#) [AWS kebijakan terkelola](#)

Secrets Manager menghasilkan entri CloudTrail log saat Anda membuat rahasia. Untuk informasi selengkapnya, lihat [the section called “Log dengan AWS CloudTrail”](#).

Untuk membuat rahasia (konsol)

1. Buka konsol Secrets Manager di <https://console.aws.amazon.com/secretsmanager/>.
2. Pilih Store a new secret (Simpan rahasia baru).
3. Pada halaman Pilih jenis rahasia, lakukan hal berikut:
  - a. Untuk tipe Rahasia, pilih jenis kredensial database yang akan disimpan:
    - Basis data Amazon RDS (termasuk Aurora)
    - Basis data Amazon DocumentDB
    - Gudang data Amazon Redshift
  - b. Untuk Kredensial, masukkan kredensial untuk database.
  - c. Untuk kunci Enkripsi, pilih Secrets Manager AWS KMS key yang digunakan untuk mengenkripsi nilai rahasia. Untuk informasi selengkapnya, lihat [Enkripsi rahasia dan dekripsi](#).
    - Untuk kebanyakan kasus, pilih aws/secretsmanager untuk menggunakan for Kunci yang dikelola AWS Secrets Manager. 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 untuk itu, pilih kunci yang dikelola pelanggan dari daftar atau pilih Tambahkan kunci baru untuk membuatnya. Untuk informasi tentang biaya penggunaan kunci yang dikelola pelanggan, lihat [Harga](#).

Anda harus memiliki [the section called “Izin untuk kunci KMS”](#). Untuk informasi tentang akses lintas akun, lihat [the section called “Akses lintas akun”](#).

- d. Untuk Database, pilih database Anda.
  - e. Pilih Selanjutnya.
4. Pada halaman Konfigurasi rahasia, lakukan hal berikut:
- a. Masukkan nama Rahasia deskriptif dan Deskripsi. Nama rahasia harus berisi 1-512 karakter Unicode.
  - b. (Opsional) Di bagian Tag, tambahkan tag ke rahasia Anda. Untuk strategi penandaan, lihat [the section called “Rahasia tag”](#). Jangan menyimpan informasi sensitif dalam tag karena tidak dienkripsi.
  - c. (Opsional) Di Izin sumber daya, untuk menambahkan kebijakan sumber daya ke rahasia Anda, pilih Edit izin. Untuk informasi selengkapnya, lihat [the section called “Lampirkan kebijakan izin ke rahasia”](#).
  - d. (Opsional) Dalam rahasia Replikasi, untuk mereplikasi rahasia Anda ke yang lain Wilayah AWS, pilih Replikasi rahasia. Anda dapat mereplikasi rahasia Anda sekarang atau kembali dan mereplikasi nanti. Untuk informasi selengkapnya, lihat [Replikasi rahasia di seluruh Wilayah](#).
  - e. 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 [Putar rahasia](#). Pilih Berikutnya.
6. Pada halaman Ulasan, tinjau detail rahasia Anda, lalu pilih Store.

Secrets Manager kembali ke daftar rahasia. Jika rahasia baru Anda tidak muncul, pilih tombol refresh.



## AWS CLI

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”](#).

Example Buat rahasia dari kredensyal dalam file JSON

[create-secret](#) Contoh berikut membuat rahasia dari kredensyal dalam file. Untuk informasi selengkapnya, lihat [Memuat AWS CLI parameter dari file](#) di Panduan AWS CLI Pengguna.

Agar Secrets Manager 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"  
}
```

## AWS SDK

Untuk membuat rahasia dengan menggunakan salah satu AWS SDK, gunakan [CreateSecret](#) tindakan. Untuk informasi selengkapnya, lihat [the section called “AWS SDK”](#).

## Struktur rahasia JSON AWS Secrets Manager

Anda dapat menyimpan teks atau biner apa pun dalam rahasia Secrets Manager. Jika Anda ingin mengaktifkan rotasi otomatis untuk rahasia Secrets Manager, itu harus dalam struktur JSON yang

benar. Selama rotasi, Secrets Manager menggunakan informasi dalam rahasia untuk terhubung ke sumber kredensi dan memperbarui kredensial di sana. Nama kunci JSON peka huruf besar/kecil.

Perhatikan bahwa ketika Anda menggunakan konsol untuk menyimpan rahasia database, Secrets Manager secara otomatis membuatnya dalam struktur JSON yang benar.

Anda dapat menambahkan lebih banyak pasangan kunci/nilai ke rahasia, misalnya dalam rahasia database, untuk memuat informasi koneksi untuk database replika di Wilayah lain.

## Topik

- [Struktur rahasia Amazon RDS Db2](#)
- [Struktur rahasia Amazon RDS MariaDB](#)
- [Amazon RDS dan Amazon Aurora MySQL struktur rahasia](#)
- [Struktur rahasia Amazon RDS Oracle](#)
- [Amazon RDS dan Amazon Aurora PostgreSQL struktur rahasia](#)
- [Amazon RDS Microsoft SQLServer struktur rahasia](#)
- [Struktur rahasia Amazon DocumentDB](#)
- [Struktur rahasia Amazon Redshift](#)
- [Amazon Redshift Struktur rahasia tanpa server](#)
- [Struktur ElastiCache rahasia Amazon](#)
- [Struktur rahasia Direktori Aktif](#)

## Struktur rahasia Amazon RDS 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": "<the ARN of the elevated secret>"
}
```

## Struktur rahasia Amazon RDS 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>
}
```

Untuk menggunakan [the section called “Pengguna bergantian”](#), Anda menyertakan `masterarn` untuk rahasia yang berisi kredensi admin atau superuser.

```
{
  "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": "<the ARN of the elevated secret>"
}
```

## Amazon RDS dan Amazon Aurora MySQL struktur rahasia

```
{
  "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>
}
```

Untuk menggunakan [the section called “Pengguna bergantian”](#), Anda menyertakan `masterarn` untuk rahasia yang berisi kredensi admin atau superuser.

```
{
  "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": "<the ARN of the elevated secret>"
}
```

## Struktur rahasia Amazon RDS Oracle

```
{
  "engine": "oracle",
  "host": "<required: instance host name/resolvable DNS name>",
  "username": "<required: username>",
  "password": "<required: password>",
  "dbname": "<required: database name>",
  "port": <optional: TCP port number. If not specified, defaults to 1521>
}
```

Untuk menggunakan [the section called “Pengguna bergantian”](#), Anda menyertakan masterarn untuk rahasia yang berisi kredensi admin atau superuser.

```
{
  "engine": "oracle",
  "host": "<required: instance host name/resolvable DNS name>",
  "username": "<required: username>",
  "password": "<required: password>",
  "dbname": "<required: database name>",
  "port": <optional: TCP port number. If not specified, defaults to 1521>,
  "masterarn": "<the ARN of the elevated secret>"
}
```

## Amazon RDS dan Amazon Aurora PostgreSQL struktur rahasia

```
{
  "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>
}
```

Untuk menggunakan [the section called “Pengguna bergantian”](#), Anda menyertakan `masterarn` untuk rahasia yang berisi kredensi admin atau superuser.

```
{
  "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": "<the ARN of the elevated secret>"
}
```

## Amazon RDS Microsoft SQLServer struktur rahasia

```
{
  "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>
}
```

Untuk menggunakan [the section called “Pengguna bergantian”](#), Anda menyertakan `masterarn` untuk rahasia yang berisi kredensi admin atau superuser.

```
{
  "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": "<the ARN of the elevated secret>"
}
```

## Struktur rahasia Amazon DocumentDB

```
{
  "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>
}

```

Untuk menggunakan [the section called “Pengguna bergantian”](#), Anda menyertakan `masterarn` untuk rahasia yang berisi kredensi admin atau superuser.

```

{
  "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>,
  "masterarn": "<the ARN of the elevated secret>",
  "ssl": <true/false. If not specified, defaults to false>
}

```

## Struktur rahasia Amazon Redshift

```

{
  "engine": "redshift",
  "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 5439>
}

```

Untuk menggunakan [the section called “Pengguna bergantian”](#), Anda menyertakan `masterarn` untuk rahasia yang berisi kredensi admin atau superuser.

```

{
  "engine": "redshift",
  "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 5439>,
"masterarn": "<the ARN of the elevated secret>"
}

```

## Amazon Redshift Struktur rahasia tanpa server

```

{
  "engine": "redshift",
  "host": "<instance host name/resolvable DNS name>",
  "username": "<username>",
  "password": "<password>",
  "dbname": "<database name. If not specified, defaults to None>",
  "namespaceName": <namespace name>,
  "port": <TCP port number. If not specified, defaults to 5439>
}

```

Untuk menggunakan [the section called “Pengguna bergantian”](#), Anda menyertakan `masterarn` untuk rahasia yang berisi kredensi admin atau superuser.

```

{
  "engine": "redshift",
  "host": "<instance host name/resolvable DNS name>",
  "username": "<username>",
  "password": "<password>",
  "dbname": "<database name. If not specified, defaults to None>",
  "namespaceName": <namespace name>,
  "port": <TCP port number. If not specified, defaults to 5439>,
  "masterarn": "<the ARN of the elevated secret>"
}

```

## Struktur ElastiCache rahasia Amazon

```

{
  "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.

## Struktur rahasia Direktori Aktif

AWS Directory Service menggunakan rahasia untuk menyimpan kredensial 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](#).

### Struktur rahasia kredensi Direktori Aktif

```
{
  "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>",
  ],
  "lastModifiedDate": "2021-07-19 17:06:58"
}
```



```
}
```

## Struktur rahasia tab tombol Direktori Aktif

Untuk informasi tentang menggunakan file tab tombol untuk mengautentikasi ke akun Active Directory di Amazon EC2, [lihat Menerapkan 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
}
```

## Buat AWS Secrets Manager rahasia

Untuk menyimpan kunci API, token akses, kredensial yang bukan untuk database, dan rahasia lainnya di Secrets Manager, ikuti langkah-langkah berikut. Untuk ElastiCache rahasia Amazon, jika Anda ingin mengaktifkan rotasi, Anda harus menyimpan rahasia dalam [struktur JSON yang diharapkan](#).

Untuk membuat rahasia, Anda memerlukan izin yang diberikan oleh [SecretsManagerReadWrite](#) [AWS kebijakan terkelola](#).

Secrets Manager menghasilkan entri CloudTrail log saat Anda membuat rahasia. Untuk informasi selengkapnya, lihat [the section called "Log dengan AWS CloudTrail"](#).

Untuk membuat rahasia (konsol)

1. Buka konsol Secrets Manager di <https://console.aws.amazon.com/secretsmanager/>.
2. Pilih Store a new secret (Simpan rahasia baru).
3. Pada halaman Pilih jenis rahasia, lakukan hal berikut:

- a. Untuk Tipe rahasia, pilih Tipe rahasia lainnya.
- b. Dalam pasangan kunci/nilai, masukkan rahasia Anda di pasangan kunci/Nilai JSON, atau pilih tab Plaintext dan masukkan rahasia dalam format apa pun. Anda dapat menyimpan hingga 65536 byte secara rahasia. Beberapa contoh:

Pasangan kunci/nilai kunci API:

**ClientID:** *my\_client\_id*

**ClientSecret** : *wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY*

Pasangan kunci/nilai kredensial:

**Username:** *saanvis*

**Password:** *CONTOH-KATA SANDI*

Teks biasa token OAuth:

*AKIAI44QH8DHBEXAMPLE*

Sertifikat digital plaintext:

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

Plaintext kunci pribadi:

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

- c. Untuk kunci Enkripsi, pilih Secrets Manager AWS KMS key yang digunakan untuk mengenkripsi nilai rahasia. Untuk informasi selengkapnya, lihat [Enkripsi rahasia dan dekripsi](#).
  - Untuk kebanyakan kasus, pilih `aws/secretsmanager` untuk menggunakan for Kunci yang dikelola AWS Secrets Manager. 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 memutarinya atau menerapkan kebijakan kunci untuk itu, pilih kunci yang dikelola pelanggan dari daftar atau pilih Tambahkan kunci baru untuk membuatnya. Untuk informasi tentang biaya penggunaan kunci yang dikelola pelanggan, lihat [Harga](#).

Anda harus memiliki [the section called “Izin untuk kunci KMS”](#). Untuk informasi tentang akses lintas akun, lihat [the section called “Akses lintas akun”](#).

- d. Pilih Selanjutnya.
4. Pada halaman Konfigurasi rahasia, lakukan hal berikut:
    - a. Masukkan nama Rahasia deskriptif dan Deskripsi. Nama rahasia harus berisi 1-512 karakter Unicode.
    - b. (Opsional) Di bagian Tag, tambahkan tag ke rahasia Anda. Untuk strategi penandaan, lihat [the section called “Rahasia tag”](#). Jangan menyimpan informasi sensitif dalam tag karena tidak dienkripsi.
    - c. (Opsional) Di Izin sumber daya, untuk menambahkan kebijakan sumber daya ke rahasia Anda, pilih Edit izin. Untuk informasi selengkapnya, lihat [the section called “Lampirkan kebijakan izin ke rahasia”](#).
    - d. (Opsional) Dalam rahasia Replikasi, untuk mereplikasi rahasia Anda ke yang lain Wilayah AWS, pilih Replikasi rahasia. Anda dapat mereplikasi rahasia Anda sekarang atau kembali dan mereplikasi nanti. Untuk informasi selengkapnya, lihat [Replikasi rahasia di seluruh Wilayah](#).
    - e. 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 [Putar rahasia](#). Pilih Berikutnya.
  6. Pada halaman Ulasan, tinjau detail rahasia Anda, lalu pilih Store.

Secrets Manager kembali ke daftar rahasia. Jika rahasia baru Anda tidak muncul, pilih tombol refresh.

## AWS CLI

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”](#).

### Example Buat rahasia

[create-secret](#) Contoh berikut menciptakan 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 dari kredensial dalam file JSON

[create-secret](#) Contoh berikut membuat rahasia dari kredensial dalam file. Untuk informasi selengkapnya, lihat [Memuat AWS CLI parameter dari file](#) di Panduan AWS CLI Pengguna.

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

Isi mycreds.json:

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

## AWS SDK

Untuk membuat rahasia dengan menggunakan salah satu AWS SDK, gunakan [CreateSecret](#) tindakan. Untuk informasi selengkapnya, lihat [the section called “AWS SDK”](#).

## Perbarui nilai untuk AWS Secrets Manager rahasia

Untuk memperbarui nilai rahasia Anda, Anda dapat menggunakan konsol, CLI, atau SDK. Saat Anda memperbarui nilai rahasia, Secrets Manager 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 Secrets Manager](#).

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 Ikhtisar, di bagian Nilai rahasia, pilih Ambil nilai rahasia dan kemudian pilih Edit.

## AWS CLI

Untuk memperbarui nilai rahasia (AWS CLI)

- 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”](#).

Berikut ini [put-secret-value](#) menciptakan versi baru dari 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 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.

Untuk memperbarui nilai rahasia, gunakan tindakan berikut: [UpdateSecret](#) atau [PutSecretValue](#). Untuk informasi selengkapnya, lihat [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:

- AWS CloudFormation — Lihat [AWS CloudFormation](#).
- 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, lihat [the section called “Versi rahasia”](#).

[update-secret-version-stage](#) Contoh berikut memindahkan label `AWSCURRENT` pementasan 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 adalah `A1b2c3d4-5678-90ab-cdef-example11111` dan versi dengan `AWSCURRENT` label adalah `A1b2c3d4-5678-90ab-cdef-example22222`. `AWSPREVIOUS` Dalam contoh ini, Anda memindahkan `AWSCURRENT` label dari versi 11111 ke 22222. Karena `AWSCURRENT` label dihapus dari versi, `update-secret-version-stage` secara otomatis

memindahkan AWSPREVIOUS label ke versi itu (11111). Efeknya adalah AWSPREVIOUS versi AWSCURRENT dan 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
```

## Ubah 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 menggunakan `aws/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 memutarinya atau menerapkan kebijakan kunci untuk itu, gunakan file. kunci yang dikelola pelanggan Anda harus memiliki [the section called “Izin untuk kunci KMS”](#). Untuk informasi tentang biaya penggunaan kunci yang dikelola pelanggan, lihat [Harga](#).

Anda dapat mengubah kunci enkripsi untuk rahasia Anda. Misalnya, jika Anda ingin [mengakses rahasia dari akun lain](#), dan rahasia saat ini dienkripsi menggunakan kunci yang AWS dikelola `aws/secretsmanager`, Anda dapat beralih ke file. kunci yang dikelola pelanggan

### Tip

Jika Anda ingin memutar kunci yang dikelola pelanggan, kami sarankan menggunakan rotasi tombol AWS KMS otomatis. Untuk informasi selengkapnya, lihat [Memutar AWS KMS tombol](#).

Saat Anda mengubah kunci enkripsi, Secrets Manager mengenkripsi ulang AWSCURRENT, AWSPENDING, 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.

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.

Jika Anda menonaktifkan kunci enkripsi sebelumnya, Anda tidak akan dapat mendekripsi versi rahasia apa pun kecuali AWSCURRENT, AWSPENDING dan AWSPREVIOUS. Jika Anda memiliki versi rahasia berlabel lain yang ingin Anda pertahankan aksesnya, Anda perlu membuat ulang versi tersebut dengan kunci enkripsi baru menggunakan [the 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 Tindakan, 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 kecuali AWSCURRENT, AWSPENDING dan AWSPREVIOUS. Jika Anda memiliki versi rahasia berlabel lain yang ingin Anda pertahankan aksesnya, Anda perlu membuat ulang versi tersebut dengan kunci enkripsi baru menggunakan [the section called “AWS CLI”](#)

Untuk mengubah kunci enkripsi untuk secret (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.



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”](#).

- a. Dapatkan nilai dari 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"
```

## Memodifikasi 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 tersebut. Atau, di AWS CLI, panggil [deskripsikan-rahasia](#), dan kemudian tinjau bidangnya. `OwningService` Untuk informasi selengkapnya, lihat [Rahasia yang dikelola](#).

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 seperangkat kredensial yang berfungsi. Untuk informasi selengkapnya, lihat [Putar rahasia](#).

Secrets Manager menghasilkan entri CloudTrail log saat Anda memodifikasi rahasia. Untuk informasi selengkapnya, lihat [the section called “Log 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 Tindakan, lalu pilih Edit deskripsi.
- Untuk memperbarui kunci enkripsi, lihat [the section called “Ubah kunci enkripsi untuk rahasia”](#).
- Untuk memperbarui tag, pada tab Tag, pilih Edit tag. Lihat [the section called “Rahasia tag”](#).
- Untuk memperbarui nilai rahasia, lihat [the section called “Perbarui nilai rahasia”](#).
- Untuk memperbarui izin rahasia Anda, pada tab Ikhtisar, pilih Edit izin. Lihat [the section called “Lampirkan kebijakan izin ke rahasia”](#).
- Untuk memperbarui rotasi rahasia Anda, pada tab Rotasi, pilih Edit rotasi. Lihat [Putar rahasia](#).
- Untuk mereplikasi rahasia Anda ke Wilayah lain, lihat [Replikasi rahasia di seluruh Wilayah](#).
- Jika rahasia Anda memiliki replika, Anda dapat mengubah kunci enkripsi untuk replika. Pada tab Replikasi, pilih tombol radio untuk replika, dan kemudian pada menu Tindakan, 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 dalam layanan itu. Lihat [Rahasia yang dikelola](#).

## 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 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.

Untuk memperbarui rahasia, gunakan tindakan berikut: [UpdateSecret](#) atau [ReplicateSecretToRegions](#). Untuk informasi selengkapnya, lihat [the section called “AWS SDK”](#).

## Temukan rahasia di AWS Secrets Manager

Saat Anda mencari rahasia tanpa filter, Secrets Manager 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, Secrets Manager menganalisis string pencarian untuk mengubahnya menjadi kata-kata terpisah. Kata-kata dipisahkan oleh perubahan dari huruf besar ke huruf kecil, dari huruf ke angka, atau dari angka/huruf ke tanda baca. Misalnya, memasukkan istilah `credsDatabase#892` pencarian mencari `creds,` dan `892` dalam `namaDatabase`, deskripsi, dan kunci tag dan nilai.

Secrets Manager menghasilkan entri CloudTrail log saat Anda mencantumkan rahasia. Untuk informasi selengkapnya, lihat [the section called “Log dengan AWS CloudTrail”](#).

Anda dapat menerapkan filter berikut ke pencarian Anda:

### Nama

Cocokkan 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: **My Description** mencocokkan 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 CyberArk atau HashiCorp.

Memiliki layanan

Cocokkan awal awalan ID layanan pengelolaan, tidak peka huruf besar/kecil. Misalnya, **my-ser** mencocokkan rahasia yang dikelola oleh layanan dengan awalan my-serv danmy-service. Untuk informasi selengkapnya, lihat [Rahasia yang dikelola](#).

Rahasia yang direplikasi

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

Tag kunci

Cocokkan awal kunci tag; peka huruf besar/kecil. Misalnya, kunci Tag: **Prod** mengembalikan rahasia dengan tag Production danProd1, tetapi bukan rahasia dengan tag prod atau1 Prod.

Nilai tag

Cocokkan awal nilai tag; peka huruf besar/kecil. Misalnya, nilai Tag: **Prod** mengembalikan rahasia dengan tag Production danProd1, tetapi bukan rahasia dengan nilai tag prod atau1 Prod.

Secrets Manager adalah layanan regional dan hanya rahasia dalam wilayah yang dipilih dikembalikan.

## 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 \  
  --filter Key="name",Values="Test"
```

Example Temukan rahasia yang dikelola oleh AWS layanan lain

[list-secrets](#) Contoh berikut mendapatkan daftar rahasia yang dikelola oleh layanan. Anda menentukan layanan dengan ID. Untuk informasi selengkapnya, lihat [Rahasia yang dikelola](#).

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

## AWS SDK

Untuk menemukan rahasia dengan menggunakan salah satu AWS SDK, gunakan [ListSecrets](#). Untuk informasi selengkapnya, lihat [the section called "AWS SDK"](#).

## Hapus AWS Secrets Manager rahasia

Karena sifat kritis rahasia, AWS Secrets Manager sengaja membuat penghapusan rahasia menjadi sulit. Secrets Manager tidak segera menghapus rahasia. Sebagai gantinya, 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. Saat Anda menghapus replika, replika segera dihapus.

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

Jika Anda tidak tahu apakah suatu aplikasi masih menggunakan rahasia, Anda dapat membuat CloudWatch alarm Amazon untuk mengingatkan Anda tentang upaya apa pun 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.

Secrets Manager menghasilkan entri CloudTrail log saat Anda menghapus rahasia. Untuk informasi selengkapnya, lihat [the section called “Log 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 Tindakan, lalu pilih Hapus rahasia.
4. Dalam kotak dialog Nonaktifkan rahasia dan jadwal penghapusan, di Masa tunggu, masukkan jumlah hari untuk menunggu sebelum penghapusan menjadi permanen. Secrets Manager melampirkan bidang yang disebut `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 Simpan.

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 Tindakan, pilih Hapus 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 Hapus 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. Untuk informasi selengkapnya, lihat [the section called "AWS SDK"](#).

## Kembalikan AWS Secrets Manager rahasia

Secrets Manager menganggap rahasia yang dijadwalkan untuk penghapusan sudah usang dan Anda tidak dapat lagi mengaksesnya secara langsung. Setelah jendela pemulihan berlalu, Secrets Manager menghapus rahasia secara permanen. Setelah Secrets Manager 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 terjadwal.

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

Secrets Manager menghasilkan entri CloudTrail log saat Anda memulihkan rahasia. Untuk informasi selengkapnya, lihat [the section called “Log 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 Simpan.

3. Pada halaman Detail rahasia, pilih Batalkan penghapusan.
4. Dalam kotak dialog Batalkan penghapusan rahasia, pilih Batalkan 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 perintah. [RestoreSecret](#) Untuk informasi selengkapnya, lihat [the section called “AWS SDK”](#).



## Tag AWS Secrets Manager rahasia

Secrets Manager mendefinisikan tag sebagai label yang terdiri dari kunci yang Anda tentukan dan nilai opsional. Anda dapat menggunakan tag untuk memudahkan mengelola, mencari, dan memfilter rahasia dan sumber daya lainnya di AWS akun Anda. Saat Anda menandai rahasia Anda, gunakan skema penamaan standar pada semua sumber daya Anda. Untuk informasi selengkapnya, lihat whitepaper [Praktik Terbaik Penandaan](#).

Anda dapat memberikan atau menolak akses ke rahasia dengan memeriksa tag yang dilampirkan pada rahasia. Untuk informasi selengkapnya, lihat [the section called “Contoh: Kontrol akses ke rahasia menggunakan tag”](#).

Anda dapat menemukan rahasia dengan tag di konsol, AWS CLI, dan SDK. AWS juga menyediakan alat [Resource Groups](#) untuk membuat konsol khusus yang menggabungkan dan mengatur sumber daya Anda berdasarkan tag mereka. Untuk menemukan rahasia dengan tag tertentu, lihat [the section called “Temukan rahasia”](#). Secrets Manager tidak mendukung alokasi biaya berbasis tag.

Jangan pernah menyimpan informasi sensitif untuk rahasia dalam tag.

Untuk kuota tag dan batasan penamaan, lihat [Kuota layanan untuk Penandaan di panduan Referensi AWS Umum](#). Tag peka terhadap huruf besar dan kecil.

Secrets Manager menghasilkan entri CloudTrail log saat Anda menandai atau menghapus tag rahasia. Untuk informasi selengkapnya, lihat [the section called “Log dengan AWS CloudTrail”](#).

Untuk mengubah tag untuk rahasia Anda (konsol)

1. Buka konsol Secrets Manager di <https://console.aws.amazon.com/secretsmanager/>.
2. Dari daftar rahasia, pilih rahasia Anda.
3. Di halaman detail rahasia, pada tab Tag, pilih Edit tag. Nama dan nilai kunci tag peka huruf besar/kecil, dan kunci tag harus unik.

## AWS CLI

Example Penambahan sebuah tag ke sebuah rahasia

[tag-resource](#) Contoh berikut menunjukkan cara melampirkan tag dengan sintaks singkatan.

```
aws secretsmanager tag-resource \
```

```
--secret-id MyTestSecret \  
--tags Key=FirstTag,Value=FirstValue
```

Example Penambahan beberapa tag ke sebuah rahasia

[tag-resource](#) Contoh berikut melampirkan dua tag kunci-nilai ke rahasia.

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

Example Menghapus tag dari sebuah 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"]'
```

## AWS SDK

Untuk mengubah tag untuk rahasia Anda, gunakan [TagResource](#) atau [UntagResource](#). Untuk informasi selengkapnya, lihat [the section called “AWS SDK”](#).

# Replikasi 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 nanti perlu, Anda dapat [mempromosikan rahasia replika ke standalone](#) dan kemudian mengaturnya untuk replikasi secara independen. Secrets Manager mereplikasi data rahasia terenkripsi dan metadata seperti tag dan kebijakan sumber daya di seluruh Wilayah tertentu.

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](#).

Ketika Anda menyimpan kredensial 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 pasangan kunci/nilai tambahan ke rahasia untuk informasi koneksi regional.

Jika Anda mengaktifkan rotasi untuk rahasia utama Anda, Secrets Manager memutar rahasia di Wilayah utama, dan nilai rahasia baru menyebar ke semua rahasia replika terkait. Anda tidak perlu mengelola rotasi satu per satu untuk semua rahasia replika.

Anda dapat mereplikasi rahasia di semua AWS Wilayah yang diaktifkan. Namun, jika Anda menggunakan Secrets Manager di AWS Wilayah khusus seperti AWS GovCloud (US) atau Wilayah China, Anda hanya dapat mengonfigurasi rahasia dan replika di dalam AWS Wilayah khusus ini. Anda tidak dapat mereplikasi rahasia di AWS Wilayah 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 [Mengelola AWS Wilayah](#).

Dimungkinkan untuk menggunakan rahasia di beberapa Wilayah tanpa mereplikasi dengan memanggil titik akhir Secrets Manager di Wilayah tempat rahasia disimpan. Untuk daftar titik

akhir, lihat [the section called “Titik akhir Secrets Manager”](#). Untuk menggunakan replikasi untuk meningkatkan ketahanan beban kerja Anda, lihat [Arsitektur Pemulihan Bencana \(DR\) di AWS, Bagian I: Strategi untuk Pemulihan](#) di Cloud.

Secrets Manager menghasilkan entri CloudTrail log saat Anda mereplikasi rahasia. Untuk informasi selengkapnya, lihat [the section called “Log 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 Replikasi rahasia.
  - Jika rahasia Anda direplikasi, di bagian Rahasia Replikasi, pilih Tambah Wilayah.
4. Dalam 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 ada di Region replika.
  - c. (Opsional) Untuk menambahkan Wilayah lain, pilih Tambahkan lebih banyak wilayah.
  - d. Pilih Replikasi.

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

## AWS CLI

Example Replikasi rahasia ke wilayah lain

[replicate-secret-to-regions](#) Contoh berikut mereplikasi rahasia eu-west-3. Replika dienkripsi dengan kunci AWS terkelola aws/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 mereplikasi ke eu-west-3. Replika dienkripsi dengan kunci AWS terkelola 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. Untuk informasi selengkapnya, lihat [the section called “AWS SDK”](#).

## Promosikan rahasia replika ke rahasia mandiri di AWS Secrets Manager

Rahasia replika adalah rahasia yang direplikasi dari primer 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 enkripsi. Mempromosikan rahasia replika memutus rahasia replika dari rahasia utama dan membuat rahasia replika 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.

Secrets Manager menghasilkan entri CloudTrail log saat Anda mempromosikan rahasia. Untuk informasi selengkapnya, lihat [the section called “Log dengan AWS CloudTrail”](#).

Untuk mempromosikan rahasia replika (konsol)

1. Masuk ke Secrets Manager 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 link 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 perintah.

[StopReplicationToReplica](#) Anda harus memanggil perintah ini dari replika Region rahasia. Untuk informasi selengkapnya, lihat [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`.

```
{
  "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

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:CreateSecret",
        "secretsmanager:ReplicateSecretToRegions"
      ],
      "Resource": "*",
      "Condition": {
        "ForAllValues:StringLike": {
          "secretsmanager:AddReplicaRegions": [
            "us-*",
            "ca-*"
          ]
        }
      }
    }
  ]
}
```

```
    }  
  }  
]  
}
```

## Memecahkan masalah replikasi AWS Secrets Manager

Berikut ini adalah beberapa alasan mengapa replikasi bisa gagal.

### Rahasia dengan nama yang sama ada di Wilayah yang dipilih

Untuk mengatasi masalah ini, Anda dapat menimpa rahasia nama duplikat di Region replika. Coba lagi replikasi, dan kemudian di kotak dialog Retry replikasi, pilih Overwrite.

### Tidak ada izin yang tersedia pada kunci KMS untuk menyelesaikan replikasi

Secrets Manager pertama kali mendekripsi rahasia sebelum mengenkripsi ulang dengan kunci KMS baru di Region 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, Secrets Manager tidak dapat mereplikasi rahasia. 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 cara Secrets Manager melakukan enkripsi, lihat [the section called “Enkripsi rahasia dan dekripsi”](#). Untuk mengatasi masalah ini, Anda dapat membuat ulang versi rahasia sehingga Secrets Manager mengenkripsi mereka dengan kunci enkripsi saat ini. Untuk informasi selengkapnya, lihat [Mengubah kunci enkripsi untuk rahasia](#). Kemudian coba lagi replikasi.

```
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 [Mengelola AWS Wilayah](#) dalam Panduan Referensi Manajemen AWS Akun.

# Dapatkan rahasia dari AWS Secrets Manager

Secrets Manager menghasilkan entri CloudTrail log saat Anda mengambil rahasia. Untuk informasi selengkapnya, lihat [the section called “Log 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 C++ SDK AWS](#)
- [Dapatkan nilai rahasia Secrets Manager menggunakan JavaScript AWS SDK](#)
- [Dapatkan nilai rahasia Secrets Manager menggunakan Kotlin AWS SDK](#)
- [Dapatkan nilai rahasia Secrets Manager menggunakan PHP AWS SDK](#)
- [Dapatkan nilai rahasia Secrets Manager menggunakan Ruby SDK AWS](#)
- [Dapatkan nilai rahasia Secrets Manager menggunakan Rust 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 AWS CloudFormation daya](#)
- [Gunakan AWS Secrets Manager rahasia di Amazon Elastic Kubernetes Service](#)
- [Gunakan AWS Secrets Manager rahasia dalam GitHub pekerjaan](#)
- [Gunakan AWS Secrets Manager rahasia di AWS IoT Greengrass](#)
- [Gunakan AWS Secrets Manager rahasia dalam AWS Lambda fungsi](#)
- [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 kredensial 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

Ketika Anda mengambil rahasia, Anda dapat menggunakan Secrets Manager komponen caching berbasis Java untuk 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, lihat [Dapatkan 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.

- AWS SDK 1.x untuk Java. Anda dapat menggunakan kedua versi AWS SDK for Java dalam proyek Anda. Untuk informasi selengkapnya, lihat [Menggunakan SDK for Java 1.x dan 2.x](#) side-by-side

Untuk mengunduh kode sumber, lihat [Secrets Manager komponen klien caching berbasis Java](#) pada 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 penanganan 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 “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, lihat [the 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 disediakan [the section called "SecretCacheConfiguration"](#).

## Metode

### getString

```
public String getString(final String secretId)
```

Mengambil rahasia string dari Secrets Manager. Mengembalikan [String](#).

### getSecretBinary

```
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. Mengembalikan true jika refresh selesai tanpa kesalahan, jika tidak false.

### tutup

```
public void close()
```

Menutup cache.

## SecretCacheConfiguration

Opsi konfigurasi cache untuk [the 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.

### getCacheHook

```
public SecretCacheHook getCacheHook()
```

Mengembalikan [the section called "SecretCacheHook"](#) antarmuka yang digunakan untuk mengaitkan pembaruan cache.

### setCacheHook

```
public void setCacheHook(SecretCacheHook cacheHook)
```

Mengatur [the section called "SecretCacheHook"](#) antarmuka yang digunakan untuk mengaitkan pembaruan cache.

### getMaxCacheUkuran

```
public int getMaxCacheSize()
```

Mengembalikan ukuran cache maksimum. Defaultnya adalah 1024 rahasia.

### setMaxCacheUkuran

```
public void setMaxCacheSize(int maxCacheSize)
```

Menetapkan ukuran cache maksimum. Defaultnya adalah 1024 rahasia.

## getCacheItemTTL

```
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. [AWSecretsManagerClient](#) Defaultnya adalah 1 jam dalam milidetik.

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

## setCacheItemTTL

```
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. [AWSecretsManagerClient](#) Defaultnya adalah 1 jam dalam milidetik.

## getVersionStage

```
public String getVersionStage()
```

Mengembalikan versi rahasia yang ingin Anda cache. Untuk informasi selengkapnya, lihat [Versi rahasia](#). Defaultnya adalah "AWSCURRENT".

## setVersionStage

```
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(AWSecretsManager client)
```

Menetapkan [AWSecretsManagerClient](#) untuk mengambil rahasia dari. Mengembalikan SecretCacheConfiguration objek diperbarui dengan pengaturan baru.

## SecretCacheConfiguration withCacheHook

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



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

#### `SecretCacheConfiguration withMaxCacheUkuran`

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

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

#### `SecretCacheConfiguration withCacheItemTTL`

```
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. [AWSecretsManagerClient](#) Defaultnya adalah 1 jam dalam milidetik. Mengembalikan `SecretCacheConfiguration` objek diperbarui dengan pengaturan baru.

#### `SecretCacheConfiguration withVersionStage`

```
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 kredensial 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 kredensial 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, lihat [the 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, dalam AWS CLI, panggilan [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 dalam rahasia sebagai pasangan kunci/nilai 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 kredensial 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:

- [Membangun 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](#)

## Membangun koneksi ke database

Contoh berikut menunjukkan cara membuat koneksi ke database menggunakan kredensial 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" ).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);
```

### Oracle

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

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

## MSSQLServer

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

## 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" ).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);
```

## Buat koneksi dengan menentukan titik akhir dan port

Contoh berikut menunjukkan cara membuat koneksi ke database menggunakan kredensial 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 pasangan kunci/nilai untuk informasi titik akhir dan 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" ).newInstance();

// 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" ).newInstance();

// 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" ).newInstance();

// 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.AWSSecretsManagerRedshiftDriver" );
```



```
// 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 kredensial 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 pasangan kunci/nilai untuk informasi titik akhir dan 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 kredensial database dalam rahasia, gunakan [driver koneksi SQL Secrets Manager](#) untuk terhubung ke database menggunakan kredensial dalam rahasia.
- Untuk jenis rahasia lainnya, gunakan [komponen caching berbasis Java Secrets Manager](#) atau panggil SDK secara langsung dengan atau. [GetSecretValueBatchGetSecretValue](#)

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 caching berbasis Secrets Manager Python 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, lihat [Dapatkan 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 berbasis komponen klien caching](#) 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 boto3
import boto3.session
from aws_secretsmanager_caching import SecretCache, SecretCacheConfig

client = boto3.session.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, lihat [the 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 dari 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) -- [Diperlukan] 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. Rahasiannya 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. Rahasiannya 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 caching berbasis Secrets Manager Python](#) atau panggil SDK secara langsung dengan atau. [get\\_secret\\_valuebatch\\_get\\_secret\\_value](#)

Contoh kode berikut menunjukkan cara menggunakan `GetSecretValue`.

Izin yang diperlukan: `secretsmanager:GetSecretValue`

```
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 memilikinyasecretsmanager:ListSecrets.

Untuk contoh kebijakan izin, lihat [the 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 .NET AWS SDK](#)

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

Saat Anda mengambil rahasia, Anda dapat menggunakan komponen caching berbasis Secrets Manager .net 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, lihat [Dapatkan 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 `GetSecretString` `GetSecretBinary` Pada pengambilan berturut-turut, cache mengembalikan salinan rahasia yang di-cache.

Untuk mendapatkan paket caching

- Lakukan salah satu hal berikut ini:
  - 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](#)
- [Aku SecretCacheHook](#)

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 Konfigurasi 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, lihat [the 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

[AmazonSecretsManagerClient](#) Untuk mengambil rahasia dari.

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

Membangun cache rahasia baru menggunakan yang disediakan [the 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 [AmazonSecretsManagerClient](#) dan 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

[AmazonSecretsManagerClient](#) Untuk 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 untuk [the section called "SecretsManagerCache"](#), seperti ukuran cache maksimum dan Time to Live (TTL) untuk rahasia cache.

### Properti

`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; }
```

[AmazonSecretsManagerClient](#) Untuk mengambil rahasia dari. Jika `null`, cache membuat instance klien baru. Nilai default-nya `null`.

### CacheHook

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

[The section called "Aku SecretCacheHook"](#).

### Aku SecretCacheHook

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 .NET 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.NET, gunakan [komponen caching berbasis Secrets Manager .NET](#) atau panggil SDK secara langsung dengan atau [GetSecretValueBatchGetSecretValue](#)

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 GetSecretValueResponse 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 caching berbasis Secrets Manager Go 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, lihat [Dapatkan 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/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, lihat [the 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 senganan 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 sengatan 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 caching berbasis Secrets Manager Go](#) atau panggil SDK secara langsung dengan atau. [GetSecretValueBatchGetSecretValue](#)

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 C++ SDK AWS

Untuk aplikasi C++, hubungi SDK secara langsung dengan [GetSecretValue](#) atau [BatchGetSecretValue](#)

Contoh kode berikut menunjukkan cara mendapatkan nilai rahasia Secrets Manager.

## 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 Secrets Manager menggunakan JavaScript AWS SDK

Untuk JavaScript aplikasi, hubungi SDK secara langsung dengan [getSecretValue](#) atau [batchGetSecretValue](#).

Contoh kode berikut menunjukkan cara mendapatkan nilai rahasia Secrets Manager.

## 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 Secrets Manager menggunakan Kotlin AWS SDK

Untuk aplikasi Kotlin, hubungi SDK secara langsung dengan [GetSecretValue](#) atau [BatchGetSecretValue](#).

Contoh kode berikut menunjukkan cara mendapatkan nilai rahasia Secrets Manager.

Izin yang diperlukan: `secretsmanager:GetSecretValue`

```
suspend fun getValue(secretName: String?) {
    val valueRequest =
        GetSecretValueRequest {
            secretId = secretName
        }

    SecretsManagerClient { region = "us-east-1" }.use { secretsClient ->
        val response = secretsClient.getSecretValue(valueRequest)
        val secret = response.secretString
        println("The secret value is $secret")
    }
}
```

## Dapatkan nilai rahasia Secrets Manager menggunakan PHP AWS SDK

Untuk aplikasi PHP, hubungi SDK langsung dengan [GetSecretValue](#) atau [BatchGetSecretValue](#).

Contoh kode berikut menunjukkan cara mendapatkan nilai rahasia Secrets Manager.

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 Secrets Manager menggunakan Ruby SDK AWS

Untuk aplikasi Ruby, hubungi SDK langsung dengan [get\\_secret\\_value](#) atau [batch\\_get\\_secret\\_value](#).

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.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
end
```

## Dapatkan nilai rahasia Secrets Manager menggunakan Rust AWS SDK

Untuk aplikasi Rust, hubungi SDK secara 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(())  
}
```

## Dapatkan nilai rahasia menggunakan AWS CLI

Izin yang diperlukan: `secretsmanager:GetSecretValue`

Example Ambil nilai rahasia terenkripsi dari sebuah 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 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 memilikinyasecretsmanager:ListSecrets.

Untuk contoh kebijakan izin, lihat [the 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 dengan 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.

Secrets Manager menampilkan versi saat ini (AWSCURRENT) dari rahasia. Untuk melihat [versi rahasia lainnya](#), seperti AWSPREVIOUS atau versi berlabel khusus, gunakan file. [the 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 AWS CloudFormation daya

Dengan AWS CloudFormation, Anda dapat mengambil rahasia untuk digunakan di AWS 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 kredensial untuk database baru. Untuk informasi tentang membuat rahasia dengan AWS CloudFormation, lihat [AWS CloudFormation](#).

Untuk mengambil rahasia dalam AWS CloudFormation template, Anda menggunakan referensi dinamis. Saat Anda membuat tumpukan, referensi dinamis menarik nilai rahasia ke AWS CloudFormation sumber daya, sehingga 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`, AWS 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. AWS CloudFormation tidak dapat menyelesaikan referensi tersebut, yang menyebabkan kegagalan sumber daya.

## Gunakan AWS Secrets Manager rahasia di Amazon Elastic Kubernetes Service

Untuk menampilkan rahasia dari Secrets Manager sebagai file yang dipasang di pod [Amazon EKS](#), Anda dapat menggunakan AWS Secrets and Configuration Provider (ASCP) untuk [Kubernetes Secrets](#) Store CSI Driver. ASCP bekerja dengan Amazon Elastic Kubernetes Service (Amazon EKS) 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 kunci/nilai dalam format JSON, Anda dapat memilih mana yang akan dipasang di Amazon EKS. ASCP menggunakan [sintaks JMESPath](#) untuk menanyakan pasangan kunci/nilai dalam rahasia Anda. ASCP juga bekerja dengan [parameter Parameter Store](#).

Jika Anda menggunakan kluster Amazon EKS pribadi, pastikan VPC tempat kluster berada memiliki titik akhir Secrets Manager. Secrets Store CSI Driver menggunakan endpoint untuk melakukan panggilan ke Secrets Manager. Untuk informasi tentang membuat endpoint di VPC, lihat [Titik akhir VPC](#)

Jika Anda menggunakan rotasi otomatis Secrets Manager untuk rahasia Anda, Anda juga dapat menggunakan fitur reconciler rotasi Driver Secrets Store CSI untuk memastikan Anda mengambil

rahasia terbaru dari Secrets Manager. Untuk informasi selengkapnya, lihat [Rotasi otomatis konten yang dipasang dan Rahasia Kubernetes yang disinkronkan](#).

## Topik

- [Langkah 1: Siapkan kontrol akses](#)
- [Langkah 2: Instal dan konfigurasi ASCP](#)
- [Langkah 3: Identifikasi rahasia mana yang akan dipasang](#)
- [Langkah 4: Pasang rahasia sebagai file di pod Amazon EKS](#)
- [Pemecahan Masalah](#)
- [SecretProviderClass](#)

## Langkah 1: Siapkan kontrol akses

ASCP mengambil identitas pod Amazon EKS 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.

Jika panggilan dari ASCP untuk mencari Region dan peran IAM yang terkait dengan pod dibatasi oleh Kubernetes, Anda dapat mengubah kuota throttling menggunakan, seperti yang ditunjukkan pada Langkah 2. `helm install`

Untuk memberikan akses pod Amazon EKS Anda 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 kluster 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 kluster 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.

## Langkah 2: Instal dan konfigurasi ASCP

ASCP tersedia GitHub di repositori [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, lihat [the section called "Titik akhir Secrets Manager"](#).

Untuk menginstal ASCP dengan menggunakan Helm

1. Untuk memastikan repo menunjuk ke grafik terbaru, gunakan `helm repo update`.
2. Tambahkan bagan Driver CSI Secrets Store.

```
helm repo add secrets-store-csi-driver https://kubernetes-sigs.github.io/secrets-store-csi-driver/charts
```

3. Instal bagan. Untuk mengonfigurasi pelambatan, tambahkan tanda berikut: `--set-json 'k8sThrottlingParams={"qps": "<number of queries per second>", "burst": "<number of queries per second>"}`

```
helm install -n kube-system csi-secrets-store secrets-store-csi-driver/secrets-store-csi-driver
```

4. Tambahkan bagan ASCP.

```
helm repo add aws-secrets-manager https://aws.github.io/secrets-store-csi-driver-provider-aws
```

5. Instal bagan. Untuk menggunakan endpoint FIPS, tambahkan tanda berikut: `--set useFipsEndpoint=true`

```
helm install -n kube-system secrets-provider-aws aws-secrets-manager/secrets-store-csi-driver-provider-aws
```

Untuk menginstal dengan menggunakan YAMG 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
```

### Langkah 3: 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”](#) SecretProviderClassDaftar rahasia untuk dipasang dan nama file untuk dipasang sebagai. SecretProviderClassHarus berada di namespace yang sama dengan pod Amazon EKS yang direferensikannya.

Contoh berikut menunjukkan cara menggunakan SecretProviderClass untuk menggambarkan rahasia yang ingin Anda pasang dan apa nama file yang dipasang di pod Amazon EKS.

Contoh:

- [Contoh: Pasang rahasia dengan nama atau ARN](#)
- [Contoh: Pasang pasangan kunci/nilai dari rahasia](#)
- [Contoh: Tentukan Wilayah failover untuk rahasia Multi-wilayah](#)
- [Contoh: Pilih rahasia failover untuk dipasang](#)

#### Contoh: Pasang rahasia dengan nama atau ARN

Contoh berikut menunjukkan SecretProviderClass yang memasang tiga file di Amazon EKS:

1. Rahasia yang ditentukan oleh ARN lengkap.
2. Rahasia yang ditentukan oleh nama.
3. Versi rahasia tertentu.

```
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-2:111122223333:secret:MySecret2-
d4e5f6"
      - objectName: "MySecret3"
        objectType: "secretsmanager"
      - objectName: "MySecret4"
        objectType: "secretsmanager"
        objectVersionLabel: "AWSCURRENT"
```

## Contoh: Pasang pasangan kunci/nilai dari rahasia

Contoh berikut menunjukkan SecretProviderClass yang memasang tiga file di Amazon EKS:

1. Rahasia yang ditentukan oleh ARN lengkap.
2. Pasangan username kunci/nilai dari rahasia yang sama.
3. Pasangan password kunci/nilai dari rahasia yang sama.

```
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-2:111122223333:secret:MySecret-
a1b2c3"
        jmesPath:
          - path: username
            objectAlias: dbusername
          - path: password
            objectAlias: dbpassword
```

## Contoh: Tentukan Wilayah failover untuk rahasia Multi-wilayah

Untuk menyediakan ketersediaan selama pemadaman konektivitas atau untuk konfigurasi pemulihan bencana, ASCP mendukung fitur failover otomatis untuk mengambil rahasia dari wilayah sekunder.

Contoh berikut menunjukkan `SecretProviderClass` yang mengambil rahasia yang direplikasi ke beberapa Wilayah. Dalam contoh ini, ASCP mencoba untuk mengambil rahasia dari keduanya `danus-east-1` dan `us-east-2`. Jika salah satu Wilayah mengembalikan kesalahan 4xx, misalnya untuk masalah otentikasi, ASCP tidak memasang salah satu rahasia. Jika rahasia berhasil diambil dari `us-east-1`, maka ASCP memasang nilai rahasia itu. Jika rahasia tidak berhasil diambil dari `us-east-1`, tetapi berhasil diambil dari `us-east-2`, maka ASCP memasang nilai rahasia itu.

```
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"
```

## Contoh: Pilih rahasia failover untuk dipasang

Contoh berikut menunjukkan `SecretProviderClass` yang menentukan rahasia mana yang akan dipasang jika terjadi failover. Rahasia failover bukanlah replika. Dalam contoh ini, ASCP mencoba untuk mengambil dua rahasia yang ditentukan oleh `objectName`. Jika salah satu mengembalikan kesalahan 4xx, misalnya untuk masalah otentikasi, ASCP tidak memasang salah satu rahasia. Jika rahasia berhasil diambil dari `us-east-1`, maka ASCP memasang nilai rahasia itu. Jika rahasia tidak berhasil diambil dari `us-east-1`, tetapi berhasil diambil dari `us-east-2`, maka ASCP memasang nilai rahasia itu. File yang dipasang di Amazon EKS diberi nama `MyMountedSecret`.

```
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-1:111122223333:secret:MySecret-
a1b2c3"
      objectAlias: "MyMountedSecret"
      failoverObject:
        - objectName: "arn:aws:secretsmanager:us-
east-2:111122223333:secret:MyFailoverSecret-d4e5f6"
```

## Langkah 4: Pasang rahasia sebagai file di pod Amazon EKS

Untuk memasang rahasia di Amazon EKS

1. Terapkan `SecretProviderClass` ke pod dengan perintah `kubectl apply -f ExampleSecretProviderClass.yaml`.
2. Terapkan pod Anda dengan perintah `kubectl apply -f ExampleDeployment.yaml`.
3. ASCP memasang file.

## 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 penyedia, dalam perintah berikut, `<PODID>`gunakan ID pod `csi-secrets-store-provider-aws`.

```
kubectl -n kube-system get pods
kubectl -n kube-system logs pod/<PODID>
```

## SecretProviderClass

Anda menggunakan YAMAL untuk menjelaskan rahasia mana yang akan [dipasang di Amazon EKS menggunakan ASCP](#). Sebagai contoh, lihat [the section called “Pasang rahasia dengan nama atau ARN”](#).

```
apiVersion: secrets-store.csi.x-k8s.io/v1
kind: SecretProviderClass
metadata:
  name: <NAME>
spec:
  provider: aws
  parameters:
    region:
    failoverRegion:
    pathTranslation:
    objects:
```

Bidang `parameters` berisi rincian permintaan pemasangan:

### region

(Opsional) Wilayah AWS Rahasiannya. Jika Anda tidak menggunakan bidang ini, ASCP mencari Wilayah dari anotasi pada node. Pencarian ini menambahkan overhead ke permintaan mount, jadi sebaiknya Anda menyediakan Region untuk cluster yang menggunakan pod dalam jumlah besar.

Jika Anda juga menentukan `failoverRegion`, ASCP mencoba untuk 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 `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.

## FailOverRegion

(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 `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, lihat [Tentukan Wilayah failover untuk rahasia Multi-wilayah](#).

## PathTranslation

(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 yang dipasang yang berisi karakter pemisah jalur. Sebagai gantinya, 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` dipasang sebagai `My_Path_Secret`.

Untuk mencegah substitusi karakter, masukkan `stringFalse`.

## objek

String yang berisi deklarasi YAML tentang rahasia yang akan dipasang. Sebaiknya gunakan karakter string atau pipe (|) multi-line YAML.

## objectName

Nama atau ARN lengkap rahasianya. Jika Anda menggunakan ARN, Anda dapat menghilangkannya. `objectType` Bidang ini menjadi nama file rahasia di pod Amazon EKS kecuali Anda menentukan `objectAlias`. Jika Anda menggunakan ARN, Wilayah di ARN harus cocok dengan bidang. `region` Jika Anda menyertakan `failoverRegion`, bidang ini mewakili primer `objectName`.

## objectType

Diperlukan jika Anda tidak menggunakan Secrets Manager ARN untuk `objectName`. Bisa salah satu `secretsmanager` atau `ssmparameter`.

## Objectalias

(Opsional) Nama file rahasia di pod Amazon EKS. Jika Anda tidak menentukan bidang ini, `objectName` muncul sebagai nama file.

## ObjectVersion

(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 `primerobjectVersion`.

## objectVersionLabel

(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 `primerobjectVersionLabel`.

## JMESPath

(Opsional) Peta kunci 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 di JSON dari nilai rahasia. Jika bidang berisi tanda hubung, gunakan tanda kutip tunggal untuk menghindarinya, misalnya: `path: "'hyphenated-path'"`

### Objectalias

Nama file yang akan dipasang di pod Amazon EKS. Jika bidang berisi tanda hubung, gunakan tanda kutip tunggal untuk menghindarinya, misalnya: `objectAlias: "'hyphenated-alias'"`

## FailOverObject

(Opsional) Jika Anda menentukan bidang ini, ASCP mencoba untuk mengambil kedua rahasia yang ditentukan dalam primer `objectName` dan rahasia yang ditentukan dalam sub-bidang `failoverObject objectName`. Jika salah satu mengembalikan kesalahan 4xx, misalnya untuk masalah otentikasi, ASCP tidak memasang salah satu rahasia. Jika rahasia berhasil diambil dari `primerobjectName`, maka ASCP memasang nilai rahasia itu. Jika rahasia tidak berhasil diambil dari `primerobjectName`, tetapi berhasil diambil dari `failoverobjectName`, maka ASCP memasang nilai rahasia itu. Jika Anda menyertakan bidang ini, Anda harus menyertakan bidang `tersebutobjectAlias`. Untuk contoh cara menggunakan bidang ini, lihat [Pilih rahasia failover untuk dipasang](#).

Anda biasanya menggunakan bidang ini ketika rahasia failover bukan replika. Untuk contoh cara menentukan replika, lihat [Tentukan Wilayah failover untuk rahasia Multi-wilayah](#).



### objectName

Nama atau ARN lengkap dari rahasia failover. Jika Anda menggunakan ARN, Wilayah di ARN harus cocok dengan bidang. `failoverRegion`

### ObjectVersion

(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.

### objectVersionLabel

(Opsional) Alias untuk versi. Defaultnya adalah versi terbaru `AWSCURRENT`. Untuk informasi selengkapnya, lihat [the section called “Versi rahasia”](#).

## 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 kredensial 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 kredensial 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 [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 rahasia JSON-parsed bernama “Secret” 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 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 perangkat Greengrass 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 inti Greengrass, sehingga konektor dan fungsi Lambda Anda dapat menggunakan rahasia lokal untuk berinteraksi dengan layanan dan aplikasi.

Untuk mengintegrasikan rahasia ke grup Greengrass, Anda membuat sumber daya grup yang mereferensi 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 dalam AWS Lambda fungsi

Anda dapat menggunakan AWS Parameter dan Rahasia Lambda Ekstensi untuk mengambil dan menyimpan rahasia AWS Secrets Manager dalam fungsi Lambda tanpa menggunakan SDK. 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. Ekstensi dapat mengambil rahasia Secrets Manager dan parameter Parameter Store. Untuk informasi tentang Parameter Store, lihat [Integrasi Parameter Store dengan ekstensi Lambda](#) di AWS Systems Manager Panduan Pengguna.

Ekstensi Lambda adalah proses pendamping yang menambah kemampuan fungsi Lambda. Untuk informasi selengkapnya, lihat [Ekstensi Lambda di Panduan Pengembang Lambda](#). Untuk informasi tentang penggunaan ekstensi dalam gambar kontainer, lihat [Bekerja dengan lapisan Lambda dan ekstensi dalam gambar kontainer](#). Lambda mencatat informasi eksekusi tentang ekstensi beserta fungsinya dengan menggunakan Amazon CloudWatch Logs. Secara default, ekstensi mencatat jumlah minimal informasi ke CloudWatch. Untuk mencatat detail lebih lanjut, atur [variabel lingkungan](#) `PARAMETERS_SECRETS_EXTENSION_LOG_LEVEL` ke `debug`.

Untuk menyediakan cache dalam memori untuk parameter dan rahasia, ekstensi mengekspos titik akhir HTTP lokal, port localhost 2773, ke lingkungan Lambda. Anda dapat mengkonfigurasi port dengan mengatur [variabel lingkungan](#) `PARAMETERS_SECRETS_EXTENSION_HTTP_PORT`.

Lambda membuat instance terpisah yang sesuai dengan tingkat konkurensi yang dibutuhkan fungsi Anda. Setiap instance diisolasi dan memelihara cache lokal sendiri dari data konfigurasi Anda. Untuk informasi selengkapnya tentang instans dan konkurensi Lambda, lihat [Mengelola konkurensi untuk fungsi Lambda di Panduan Pengembang Lambda](#).

Untuk menambahkan ekstensi untuk ARM, Anda harus menggunakan `arm64` arsitektur untuk fungsi Lambda Anda. Untuk informasi selengkapnya, lihat [Arsitektur set instruksi Lambda di Panduan Pengembang Lambda](#). Ekstensi ini mendukung ARM di Wilayah berikut: Asia Pasifik (Mumbai), AS Timur (Ohio), Eropa (Irlandia), Eropa (Frankfurt), Eropa (Zurich), AS Timur (Virginia N.), Eropa (London), Eropa (Spanyol), Asia Pasifik (Tokyo), AS Barat (Oregon), Asia Pasifik (Singapura), Asia Pasifik (Hyderabad), dan Asia Pasifik (Sydney).

Ekstensi menggunakan AWS klien. Untuk informasi tentang mengonfigurasi AWS klien, lihat [Referensi pengaturan](#) di AWS SDK dan Panduan Referensi Alat. 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 [Titik akhir VPC](#).

Izin yang diperlukan:

- [Peran eksekusi](#) Lambda harus memiliki `secretsmanager:GetSecretValue` izin untuk rahasia.
- Jika rahasia dienkripsi dengan kunci yang dikelola pelanggan alih-alih Kunci yang dikelola AWS `aws/secretsmanager`, peran eksekusi juga memerlukan `kms:Decrypt` izin untuk kunci KMS.

Untuk menggunakan AWS Parameter dan Rahasia Ekstensi Lambda

1. Tambahkan AWS layer bernama AWS Parameter dan Rahasia Lambda Extension ke fungsi Anda. Untuk petunjuk, lihat [Menambahkan lapisan ke fungsi](#) di Panduan Pengembang Lambda.



Jika Anda menggunakan AWS CLI untuk menambahkan layer, Anda memerlukan ARN ekstensi. Untuk daftar ARN, lihat [AWS Parameter dan Rahasia Lambda Extension](#) ARN di AWS Systems Manager Panduan Pengguna.

2. Berikan izin ke peran [eksekusi Lambda](#) untuk dapat mengakses rahasia:
  - `secretsmanager:GetSecretValue` izin untuk rahasia. Lihat [the section called “Contoh: Izin untuk mengambil nilai rahasia individu”](#).
  - (Opsional) Jika rahasia dienkripsi dengan kunci yang dikelola pelanggan alih-alih Kunci yang dikelola AWS `aws/secretsmanager`, peran eksekusi juga memerlukan `kms:Decrypt` izin untuk kunci KMS.
  - Anda dapat menggunakan Attribute Based Access Control (ABAC) dengan peran Lambda untuk memungkinkan akses yang lebih terperinci ke rahasia di akun. Untuk informasi selengkapnya, lihat [the section called “Contoh: Kontrol akses ke rahasia menggunakan tag”](#) dan [the section called “Contoh: Batasi akses ke identitas dengan tag yang cocok dengan tag rahasia”](#).
3. Konfigurasi cache dengan variabel [lingkungan](#) Lambda.
4. Untuk mengambil rahasia dari cache ekstensi, Anda harus terlebih dahulu menambahkan `X-AWS-Parameters-Secrets-Token` ke header permintaan. Setel token ke `AWS_SESSION_TOKEN`, yang disediakan oleh Lambda untuk semua fungsi yang berjalan. Menggunakan header ini menunjukkan bahwa penelepon berada dalam lingkungan Lambda.

Contoh Python berikut menunjukkan bagaimana menambahkan header.

```
import os
headers = {"X-Aws-Parameters-Secrets-Token": os.environ.get('AWS_SESSION_TOKEN')}
```

5. Untuk mengambil rahasia dalam fungsi Lambda, gunakan salah satu permintaan HTTP GET berikut:
  - Untuk mengambil rahasia, `secretId`, gunakan ARN atau nama rahasia.

```
GET: /secretsmanager/get?secretId=secretId
```

- Untuk mengambil nilai rahasia sebelumnya atau versi tertentu dengan label pementasan, `secretId`, gunakan ARN atau nama rahasia, dan `versionStage`, gunakan label pementasan.

```
GET: /secretsmanager/get?secretId=secretId&versionStage=AWSPREVIOUS
```

- Untuk mengambil versi rahasia tertentu berdasarkan ID, untuksecretId, gunakan ARN atau nama rahasia, dan versionId untuk, gunakan ID versi.

```
GET: /secretsmanager/get?secretId=secretId&versionId=versionId
```

### Example Mengambil rahasia (Python)

Contoh Python berikut menunjukkan bagaimana untuk mengambil rahasia dan mengurai hasil menggunakan [json.loads](#)

```
secrets_extension_endpoint = "http://localhost:" + \  
    secrets_extension_http_port + \  
    "/secretsmanager/get?secretId=" + \  
    <secret_name>  
  
r = requests.get(secrets_extension_endpoint, headers=headers)  
  
secret = json.loads(r.text)["SecretString"] # load the Secrets Manager response  
into a Python dictionary, access the secret
```

## AWS Parameter dan Rahasia variabel lingkungan Ekstensi Lambda

Anda dapat mengkonfigurasi ekstensi dengan variabel lingkungan berikut.

Untuk informasi tentang cara menggunakan variabel lingkungan, lihat [Menggunakan variabel lingkungan Lambda di Panduan Pengembang](#) Lambda.

### PARAMETERS\_SECRETS\_EXTENSION\_CACHE\_ENABLED

Setel ke true ke parameter cache dan rahasia. Setel ke false tanpa caching. Default adalah benar.

### PARAMETERS\_SECRETS\_EXTENSION\_CACHE\_SIZE

Jumlah maksimum rahasia dan parameter untuk cache. Harus berupa nilai dari 0 hingga 1000. Nilai 0 berarti tidak ada caching. Variabel ini diabaikan jika SECRETS\_MANAGER\_TTL keduanya SSM\_PARAMETER\_STORE\_TTL dan 0. Defaultnya adalah 1000.

### PARAMETERS\_SECRETS\_EXTENSION\_HTTP\_PORT

Port untuk server HTTP lokal. Defaultnya adalah 2773.

## PARAMETERS\_SECRETS\_EXTENSION\_LOG\_LEVEL

Tingkat pencatatan ekstensi menyediakan: `debug`, `info`, `warn`, `error`, atau `none`. Setel `debug` untuk melihat konfigurasi cache. Default-nya adalah `info`.

## PARAMETERS\_SECRETS\_EXTENSION\_MAX\_CONNECTIONS

Jumlah maksimum koneksi untuk klien HTTP yang digunakan ekstensi untuk membuat permintaan ke Parameter Store atau Secrets Manager. Ini adalah konfigurasi per klien. Defaultnya adalah 3.

## SECRETS\_MANAGER\_TIMEOUT\_MILLIS

Batas waktu untuk permintaan ke Secrets Manager dalam milidetik. Nilai 0 berarti tidak ada batas waktu. Default-nya adalah 0.

## SECRETS\_MANAGER\_TTL

TTL rahasia dalam cache dalam hitungan detik. Nilai 0 berarti tidak ada caching. Maksimumnya adalah 300 detik. Variabel ini diabaikan `PARAMETERS_SECRETS_CACHE_SIZE` jika 0. Defaultnya adalah 300 detik.

## SSM\_PARAMETER\_STORE\_TIMEOUT\_MILLIS

Batas waktu untuk permintaan ke Parameter Store dalam milidetik. Nilai 0 berarti tidak ada batas waktu. Default-nya adalah 0.

## SSM\_PARAMETER\_STORE\_TTL

TTL parameter dalam cache dalam hitungan detik. Nilai 0 berarti tidak ada caching. Maksimumnya adalah 300 detik. Variabel ini diabaikan `PARAMETERS_SECRETS_CACHE_SIZE` jika 0. Defaultnya adalah 300 detik.

## 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.
- [the section called “Rotasi oleh 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, di mana layanan mengkonfigurasi dan mengelola rotasi 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 master. 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 dengan Service Connect](#) di Panduan Pengembang Layanan Amazon Elastic Container.
- Amazon RDS menawarkan rotasi terkelola untuk kredensial pengguna master. Untuk informasi selengkapnya, lihat [Manajemen kata sandi dengan Amazon RDS dan AWS Secrets Manager di Panduan Pengguna Amazon RDS](#).
- Amazon Redshift menawarkan rotasi terkelola untuk kata sandi admin. Untuk informasi selengkapnya, lihat [Mengelola kata sandi admin Amazon Redshift menggunakan AWS Secrets Manager](#) dalam Panduan Manajemen Amazon Redshift.

**Tip**

Untuk semua jenis rahasia lainnya, lihat [the section called “Rotasi oleh 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 utama. Untuk pengguna aplikasi, untuk ketersediaan tertinggi, Anda dapat menggunakan [strategi rotasi pengguna Alternating](#).

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 Secrets Manager.
2. 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 memutar rahasia sesering setiap empat jam. Untuk informasi selengkapnya, lihat [Jadwal rotasi](#).
3. (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.
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 pertama dan ke-15 setiap bulan. Untuk informasi selengkapnya, lihat [Jadwal rotasi](#).

```
aws secretsmanager rotate-secret \  
  --secret-id MySecret \  
  --rotation-rules "{\"ScheduleExpression\": \"cron(0 16 1,15 * ? *)\", \  
  \"Duration\": \"2h\"}"
```

## Rotasi oleh 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 beberapa [Rahasia yang dikelola](#), Anda menggunakan rotasi terkelola. Untuk menggunakan [Rotasi 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 “Log 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"
}
```

Jika ada langkah rotasi yang gagal, Secrets Manager mencoba ulang seluruh proses rotasi beberapa kali.

### 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](#)
- [Mengatur 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 rotasi Lambda](#)
- [Memecahkan masalah rotasi AWS Secrets Manager](#)

## Siapkan rotasi otomatis untuk rahasia Amazon RDS, Amazon Aurora, Amazon Redshift, atau Amazon DocumentDB

Tutorial ini menjelaskan cara mengatur [the section called “Rotasi oleh fungsi Lambda”](#) rahasia database. Rotasi adalah proses memperbarui rahasia secara berkala. Ketika Anda memutar rahasia, Anda memperbarui kredensial 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: Konfigurasi 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, lihat [the section called “Strategi rotasi fungsi Lambda”](#).

Jika Anda memilih strategi pengguna bergantian, Anda harus [Buat rahasia database](#) dan menyimpan kredensial superuser database di dalamnya. Anda memerlukan rahasia dengan kredensial 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: Konfigurasi 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 memutar rahasia sesering 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 Anda

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 memutar.

Untuk memperbarui peran eksekusi fungsi rotasi Anda

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 rotasi Lambda”](#).

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 oleh 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, lihat [Rotasi 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: Konfigurasi 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 menuliskannya di langkah selanjutnya. Untuk informasi tentang cara kerja fungsi rotasi, lihat [the 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 hal berikut ini:
  - 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, konfigurasi, 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, lihat [the section called “Titik akhir Secrets Manager”](#).
    - Untuk menempatkan fungsi Lambda dalam VPC, sertakan Id dan. vpcSecurityGroup  
vpcSubnetIds
  - 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 3.9.
      - iv. Pilih Buat fungsi.

## Langkah 2: Tulis kode fungsi rotasi

Pada langkah ini, Anda menulis kode yang memperbarui rahasia dan layanan atau database yang menjadi rahasia itu. Untuk informasi tentang fungsi rotasi, termasuk tips menulis fungsi rotasi Anda sendiri, lihat [the section called “Fungsi rotasi Lambda”](#). Anda juga dapat menggunakan [Templat fungsi rotasi](#) sebagai referensi.

## Langkah 3: Konfigurasi 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 memutar rahasia sesering 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, lihat [izin 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.

```
{
  "Version": "2012-10-17",
  "Id": "default",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "secretsmanager.amazonaws.com"
      },
      "Action": "lambda:InvokeFunction",
      "Condition": {
        "StringEquals": {
          "AWS:SourceAccount": "123456789012"
        }
      },
      "Resource": "LambdaRotationFunctionARN"
    }
  ]
}
```

## 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 rotasi Lambda”](#).

## 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 di [Putar rahasia segera](#).

Jika rotasi gagal, lihat [Memecahkan masalah rotasi](#).

## Mengatur rotasi otomatis menggunakan AWS CLI

Tutorial ini menjelaskan cara mengatur [the section called “Rotasi oleh fungsi Lambda”](#) dengan menggunakan AWS CLI. Ketika Anda memutar rahasia, Anda memperbarui kredensi di kedua rahasia dan database atau layanan yang menjadi rahasia itu.

Anda juga dapat mengatur rotasi menggunakan konsol. Untuk rahasia database, lihat [Rotasi otomatis untuk rahasia database \(konsol\)](#). Untuk semua jenis rahasia lainnya, lihat [Rotasi 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 kredensi untuk superuser database. Selanjutnya, Anda menulis kode fungsi rotasi. Secrets Manager menyediakan template tempat Anda dapat 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 Secrets Manager 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: Konfigurasi rahasia untuk rotasi](#)



- [Langkah selanjutnya](#)

## Prasyarat untuk rahasia database: Pilih strategi rotasi

Untuk informasi tentang strategi yang ditawarkan oleh Secrets Manager, lihat [the 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 database](#) 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 oleh fungsi Lambda”](#). Pada langkah ini, Anda menulis kode yang memperbarui rahasia dan layanan atau basis data yang menjadi tujuan rahasianya.

Secrets Manager menyediakan template untuk Amazon RDS, Amazon Aurora, Amazon Redshift, dan rahasia database Amazon DocumentDB di [Templat fungsi rotasi](#)

Untuk menulis kode fungsi rotasi

1. Lakukan salah satu hal berikut ini:
  - Periksa daftar [templat fungsi rotasi](#). Jika ada yang cocok dengan layanan dan strategi 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, lihat [the section called “Izin untuk rotasi”](#). Kebijakan harus:
  - Izinkan peran untuk memanggil operasi Secrets Manager pada rahasia.
  - Izinkan peran untuk memanggil layanan yang rahasianya, misalnya, untuk membuat kata sandi baru.
2. Buat peran eksekusi Lambda dan terapkan kebijakan kepercayaan yang Anda buat di langkah sebelumnya dengan menelepon. [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 menelepon. [lambda create-function](#)

```
aws lambda create-function \  
  --function-name my-rotation-function \  
  --runtime python3.7 \  
  --zip-file fileb://my-function.zip \  
  --handler .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 menelepon. [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 rotasi Lambda”](#).

### Langkah 4: Konfigurasi rahasia untuk rotasi

Untuk mengaktifkan rotasi otomatis untuk rahasia Anda, hubungi [rotate-secret](#). Anda dapat mengatur jadwal rotasi dengan ekspresi `cron()` atau `rate()` jadwal, 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: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

Untuk [the section called “Rotasi oleh 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 kredensial yang diputar. Anda dapat mengurangi risiko ini dengan strategi coba [lagi yang tepat](#). Setelah rotasi, koneksi baru menggunakan kredensial baru.

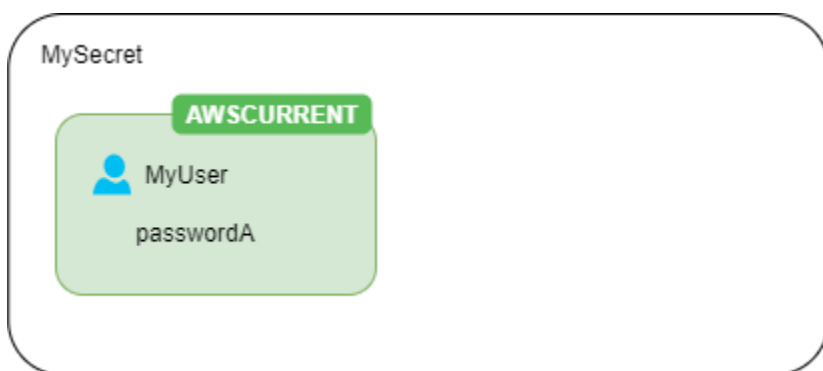
## Strategi rotasi: pengguna bergantian

Strategi ini memperbarui kredensial untuk dua pengguna dalam satu rahasia. Anda membuat pengguna pertama, dan selama rotasi pertama, fungsi rotasi mengkloningnya untuk membuat pengguna kedua. Setiap kali rahasia berputar, fungsi rotasi mengganti kata sandi pengguna mana yang diperbarui. 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 kredensial untuk pengguna satu kali (ad hoc) atau interaktif.

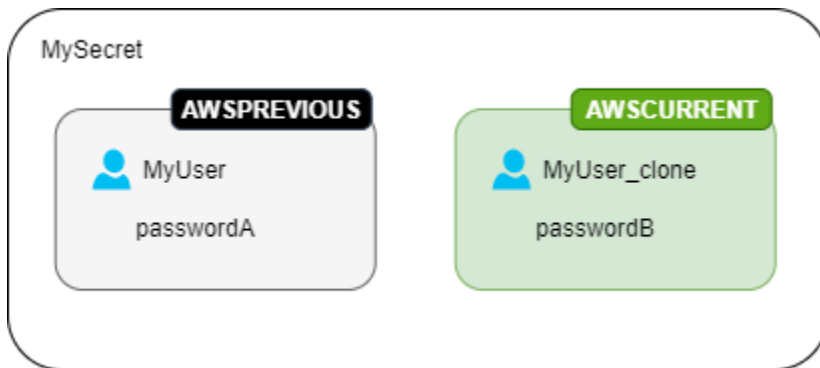
Strategi ini sesuai untuk database dengan model izin di mana satu peran memiliki tabel database dan peran kedua memiliki izin untuk mengakses tabel database. Ini juga sesuai untuk aplikasi yang membutuhkan ketersediaan tinggi. Jika aplikasi mengambil rahasia selama rotasi, aplikasi masih mendapatkan set kredensial 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 kredensial 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.

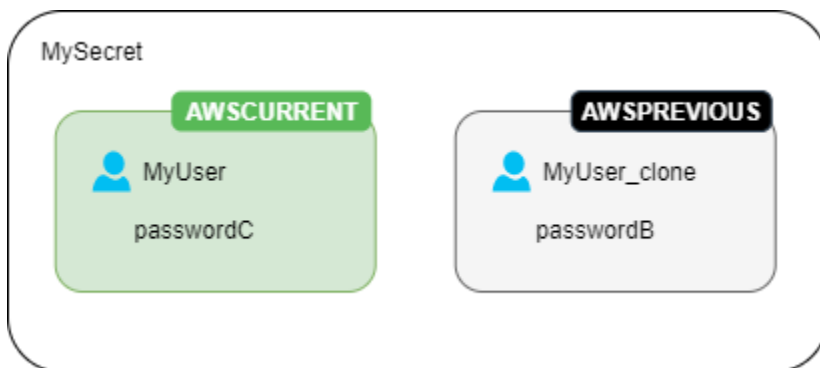
Misalnya, jika Anda membuat rahasia dengan kredensial pengguna database, rahasia berisi satu versi dengan kredensialnya.



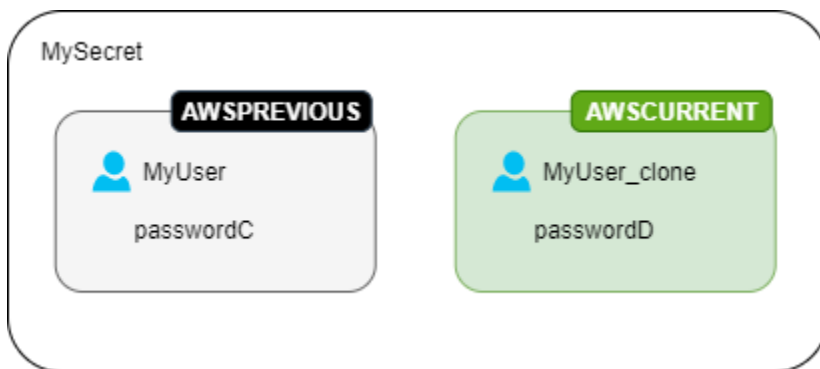
Rotasi pertama - Fungsi rotasi membuat tiruan pengguna Anda dengan kata sandi yang dihasilkan, dan kredensi tersebut menjadi versi rahasia saat ini.



Rotasi kedua - Fungsi rotasi memperbarui kata sandi untuk pengguna asli.



Rotasi ketiga - Fungsi rotasi memperbarui kata sandi untuk pengguna kloning.



## Fungsi rotasi Lambda

Dalam [the section called "Rotasi oleh fungsi Lambda"](#), fungsi Lambda melakukan pekerjaan memutar rahasia. Secrets Manager menggunakan [label pementasan](#) untuk memberi label versi rahasia selama rotasi.

Jika Secrets Manager tidak menyediakan [template fungsi rotasi](#) untuk jenis rahasia Anda, Anda dapat membuat fungsi rotasi. Saat menulis fungsi rotasi, ikuti panduan untuk setiap langkah.

Kiat untuk menulis fungsi rotasi Anda sendiri

- Gunakan [template rotasi generik](#) sebagai titik awal untuk menulis fungsi rotasi Anda sendiri.
- Saat Anda menulis fungsi Anda, berhati-hatilah untuk menyertakan pernyataan debugging atau logging. Pernyataan ini dapat menyebabkan informasi dalam fungsi Anda ditulis ke Amazon CloudWatch, jadi Anda perlu memastikan log tidak menyertakan informasi sensitif yang dikumpulkan selama pengembangan.

Untuk contoh pernyataan log, lihat kode [the section called “Templat fungsi rotasi”](#) sumber.

- Untuk keamanan, Secrets Manager hanya mengizinkan fungsi rotasi Lambda untuk memutar rahasia secara langsung. Fungsi rotasi tidak dapat memanggil fungsi Lambda kedua untuk memutar rahasia.
- Untuk saran debugging, lihat [Menguji dan men-debug aplikasi tanpa server](#).
- Jika Anda menggunakan binari dan pustaka eksternal, misalnya untuk terhubung ke sumber daya, Anda harus mengelola tambalan dan menyimpannya. up-to-date
- Simpan fungsi rotasi Anda dalam file ZIP *my-function.zip* bersama dengan dependensi yang diperlukan.

Empat langkah dalam fungsi rotasi

Topik

- [create\\_secret: Buat versi baru dari rahasia](#)
- [set\\_secret: Ubah kredensial dalam database atau layanan](#)
- [test\\_secret: Uji versi rahasia baru](#)
- [finish\\_secret: Selesaikan rotasi](#)

**create\_secret:** Buat versi baru dari rahasia

Metode `create_secret` 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.

**set\_secret**: Ubah kredensial dalam database atau layanan

Metode ini `set_secret` 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 kredensial 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.

### **test\_secret:** 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.

### **finish\_secret:** Selesaikan rotasi

Terakhir, fungsi rotasi Lambda memindahkan label AWSCURRENT dari versi rahasia sebelumnya ke versi ini, yang juga menghapus AWSPENDING label dalam panggilan API yang sama. 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. Untuk informasi selengkapnya, lihat [Memecahkan masalah rotasi](#).

## AWS Secrets Manager templat fungsi rotasi

Untuk [the section called “Rotasi oleh fungsi Lambda”](#), Secrets Manager menyediakan sejumlah template fungsi rotasi. Untuk menggunakan template, lihat:

- [Rotasi otomatis untuk rahasia database \(konsol\)](#)
- [Rotasi otomatis untuk rahasia non-database \(konsol\)](#)

Template mendukung Python 3.9.

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 ElastiCache](#)
- [Direktori Aktif](#)
  - [Kredensial Direktori Aktif](#)
  - [Tab tombol Direktori Aktif](#)
- [Jenis rahasia lainnya](#)

## Amazon RDS dan Amazon Aurora

### Amazon RDS Db2 pengguna tunggal

- Nama templat: SecretsManager RDSdB2 RotationSingleUser
- Strategi rotasi: [Strategi rotasi: pengguna tunggal](#).
- **SecretString**struktur: [the section called “Struktur rahasia Amazon RDS Db2”](#).
- Kode sumber: [https://github.com/aws-samples/ aws-secrets-manager-rotation -lambdas/tree/ master/ SecretsManager RDSdB2 /lambda\\_function.py](https://github.com/aws-samples/aws-secrets-manager-rotation-lambdas/tree/master/SecretsManager%20RDSdB2/lambda_function.py) RotationSingleUser
- Ketergantungan: [python-ibmdb](#)

### Amazon RDS Db2 bergantian pengguna

- Nama templat: SecretsManager RDSdB2 RotationMultiUser
- Strategi rotasi: [the section called “Pengguna bergantian”](#).
- **SecretString**struktur: [the section called “Struktur rahasia Amazon RDS Db2”](#).
- Kode sumber: [https://github.com/aws-samples/ aws-secrets-manager-rotation -lambdas/tree/ master/ SecretsManager RDSdB2 /lambda\\_function.py](https://github.com/aws-samples/aws-secrets-manager-rotation-lambdas/tree/master/SecretsManager%20RDSdB2/lambda_function.py) RotationMultiUser
- Ketergantungan: [python-ibmdb](#)

### Amazon RDS MariaDB pengguna tunggal

- Nama template: SecretsManager RDSmariaDB RotationSingleUser
- Strategi rotasi: [Strategi rotasi: pengguna tunggal](#).
- **SecretString**struktur: [the section called “Struktur rahasia Amazon RDS MariaDB”](#).

- Kode sumber: [https://github.com/aws-samples/ aws-secrets-manager-rotation -lambdas/tree/ master/ RDSmariaDB /lambda\\_function.py SecretsManager RotationSingleUser](https://github.com/aws-samples/aws-secrets-manager-rotation-lambdas/tree/master/RDSmariaDB/lambda_function.py)
- Ketergantungan: PyMy SQL 1.0.2. 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: SecretsManager RDSmariaDB RotationMultiUser
- Strategi rotasi:[the section called “Pengguna bergantian”](#).
- **SecretString**struktur:[the section called “Struktur rahasia Amazon RDS MariaDB”](#).
- Kode sumber: [https://github.com/aws-samples/ aws-secrets-manager-rotation -lambdas/tree/ master/ RDSmariaDB /lambda\\_function.py SecretsManager RotationMultiUser](https://github.com/aws-samples/aws-secrets-manager-rotation-lambdas/tree/master/RDSmariaDB/lambda_function.py)
- Ketergantungan: PyMy SQL 1.0.2. 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: SecretsManager RdsMySQL RotationSingleUser
- Strategi rotasi:[the section called “Pengguna tunggal”](#).
- **SecretString**Struktur yang diharapkan:[the section called “Amazon RDS dan Amazon Aurora MySQL struktur rahasia”](#).
- Kode sumber: [https://github.com/aws-samples/ aws-secrets-manager-rotation -lambdas/tree/ master/ SecretsManager RDSMySQL /lambda\\_function.py RotationSingleUser](https://github.com/aws-samples/aws-secrets-manager-rotation-lambdas/tree/master/SecretsManagerRDSMySQL/lambda_function.py)
- Ketergantungan: PyMy SQL 1.0.2. 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: SecretsManager RdsMySQL RotationMultiUser
- Strategi rotasi:[the section called “Pengguna bergantian”](#).
- **SecretString**Struktur yang diharapkan:[the section called “Amazon RDS dan Amazon Aurora MySQL struktur rahasia”](#).
- Kode sumber: [https://github.com/aws-samples/ aws-secrets-manager-rotation -lambdas/tree/ master/ SecretsManager RDSMySQL /lambda\\_function.py](https://github.com/aws-samples/aws-secrets-manager-rotation-lambdas/tree/master/SecretsManagerRDSMySQL/lambda_function.py) RotationMultiUser
- Ketergantungan: PyMy SQL 1.0.2. 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: SecretsManager RDS OracleRotationSingleUser
- Strategi rotasi:[the section called “Pengguna tunggal”](#).
- **SecretString**Struktur yang diharapkan:[the section called “Struktur rahasia Amazon RDS Oracle”](#).
- Kode sumber: [https://github.com/aws-samples/ aws-secrets-manager-rotation SecretsManager - lambdas/tree/master/ RDS /lambda\\_function.py](https://github.com/aws-samples/aws-secrets-manager-rotation-lambdas/tree/master/RDS/lambda_function.py) OracleRotationSingleUser
- Ketergantungan: [python-oracledb](#) 2.0.1

## Amazon RDS Oracle bergantian pengguna

- Nama template: SecretsManager RDS OracleRotationMultiUser
- Strategi rotasi:[the section called “Pengguna bergantian”](#).
- **SecretString**Struktur yang diharapkan:[the section called “Struktur rahasia Amazon RDS Oracle”](#).
- Kode sumber: [https://github.com/aws-samples/ aws-secrets-manager-rotation SecretsManager - lambdas/tree/master/ RDS /lambda\\_function.py](https://github.com/aws-samples/aws-secrets-manager-rotation-lambdas/tree/master/RDS/lambda_function.py) OracleRotationMultiUser
- Ketergantungan: [python-oracledb](#) 2.0.1

## Amazon RDS dan Amazon Aurora PostgreSQL pengguna tunggal

- Nama template: SecretsManager RDSPostgreSQL RotationSingleUser
- Strategi rotasi:[Strategi rotasi: pengguna tunggal](#).
- **SecretString**Struktur yang diharapkan:[the section called “Amazon RDS dan Amazon Aurora PostgreSQL struktur rahasia”](#).
- Kode sumber: [https://github.com/aws-samples/ aws-secrets-manager-rotation -lambdas/tree/ master/ SecretsManager RDSPostgreSQL /lambda\\_function.py](https://github.com/aws-samples/aws-secrets-manager-rotation-lambdas/tree/master/SecretsManager%20RDSPostgreSQL) RotationSingleUser
- Ketergantungan: PyGre SQL 5.0.7

## Amazon RDS dan Amazon Aurora PostgreSQL pengguna bergantian

- Nama template: SecretsManager RDSPostgreSQL RotationMultiUser
- Strategi rotasi:[the section called “Pengguna bergantian”](#).
- **SecretString**Struktur yang diharapkan:[the section called “Amazon RDS dan Amazon Aurora PostgreSQL struktur rahasia”](#).
- Kode sumber: [https://github.com/aws-samples/ aws-secrets-manager-rotation -lambdas/tree/ master/ SecretsManager RDSPostgreSQL /lambda\\_function.py](https://github.com/aws-samples/aws-secrets-manager-rotation-lambdas/tree/master/SecretsManager%20RDSPostgreSQL) RotationMultiUser
- Ketergantungan: PyGre SQL 5.0.7

## Amazon RDS Microsoft SQLServer pengguna tunggal

- Nama template: SecretsManager RDSSQL ServerRotationSingleUser
- Strategi rotasi:[the section called “Pengguna tunggal”](#).
- **SecretString**Struktur yang diharapkan:[the section called “Amazon RDS Microsoft SQLServer struktur rahasia”](#).
- Kode sumber: [https://github.com/aws-samples/ aws-secrets-manager-rotation -lambdas/tree/ master/ SecretsManager RDSSQL /lambda\\_function.py](https://github.com/aws-samples/aws-secrets-manager-rotation-lambdas/tree/master/SecretsManager%20RDSSQL) ServerRotationSingleUser
- Ketergantungan: Pymssql 2.2.2

## Amazon RDS Microsoft SQLServer bergantian pengguna

- Nama template: SecretsManager RDSSQL ServerRotationMultiUser
- Strategi rotasi:[the section called “Pengguna bergantian”](#).

- **SecretString** Struktur yang diharapkan: [the section called “Amazon RDS Microsoft SQL Server struktur rahasia”](#).
- Kode sumber: [https://github.com/aws-samples/ aws-secrets-manager-rotation -lambdas/tree/ master/ SecretsManager RDSSQL /lambda\\_function.py](https://github.com/aws-samples/aws-secrets-manager-rotation-lambdas/tree/master/SecretsManagerRDSSQL/lambda_function.py) ServerRotationMultiUser
- Ketergantungan: Pymssql 2.2.2

## Amazon DocumentDB (dengan kompatibilitas MongoDB)

### Amazon DocumentDB pengguna tunggal

- Nama template: SecretsManagerMongo DB RotationSingleUser
- Strategi rotasi: [the section called “Pengguna tunggal”](#).
- **SecretString** Struktur yang diharapkan: [the section called “Struktur rahasia Amazon DocumentDB”](#).
- Kode sumber: [https://github.com/aws-samples/ aws-secrets-manager-rotation SecretsManagerMongo -lambdas/pohon/master/ DB /lambda\\_function.py](https://github.com/aws-samples/aws-secrets-manager-rotation/SecretsManagerMongo-lambdas/pohon/master/DB/lambda_function.py) RotationSingleUser
- Ketergantungan: Pymongo 3.2

### Amazon DocumentDB pengguna bergantian

- Nama template: SecretsManagerMongo DB RotationMultiUser
- Strategi rotasi: [the section called “Pengguna bergantian”](#).
- **SecretString** Struktur yang diharapkan: [the section called “Struktur rahasia Amazon DocumentDB”](#).
- Kode sumber: [https://github.com/aws-samples/ aws-secrets-manager-rotation SecretsManagerMongo -lambdas/pohon/master/ DB /lambda\\_function.py](https://github.com/aws-samples/aws-secrets-manager-rotation/SecretsManagerMongo-lambdas/pohon/master/DB/lambda_function.py) RotationMultiUser
- Ketergantungan: Pymongo 3.2

## Amazon Redshift

### Amazon Redshift pengguna tunggal

- Nama template: SecretsManagerRedshiftRotationSingleUser
- Strategi rotasi: [the section called “Pengguna tunggal”](#).

- **SecretString** Struktur yang diharapkan: [the section called “Struktur rahasia Amazon Redshift”](#) atau [the section called “Amazon Redshift Struktur rahasia tanpa server”](#).
- Kode sumber: [https://github.com/aws-samples/ aws-secrets-manager-rotation -lambdas/tree/ master/ /lambda\\_function.py](https://github.com/aws-samples/aws-secrets-manager-rotation-lambdas/tree/master/lambda_function.py) SecretsManagerRedshiftRotationSingleUser
- Ketergantungan: PyGre SQL 5.0.7

### Amazon Redshift bergantian pengguna

- Nama template: SecretsManagerRedshiftRotationMultiUser
- Strategi rotasi: [the section called “Pengguna bergantian”](#).
- **SecretString** Struktur yang diharapkan: [the section called “Struktur rahasia Amazon Redshift”](#) atau [the section called “Amazon Redshift Struktur rahasia tanpa server”](#).
- Kode sumber: [https://github.com/aws-samples/ aws-secrets-manager-rotation -lambdas/tree/ master/ /lambda\\_function.py](https://github.com/aws-samples/aws-secrets-manager-rotation-lambdas/tree/master/lambda_function.py) SecretsManagerRedshiftRotationMultiUser
- Ketergantungan: PyGre SQL 5.0.7

### 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 “Struktur ElastiCache rahasia Amazon”](#).
- Kode sumber: [https://github.com/aws-samples/ aws-secrets-manager-rotation -lambdas/tree/ master/ /lambda\\_function.py](https://github.com/aws-samples/aws-secrets-manager-rotation-lambdas/tree/master/lambda_function.py) SecretsManagerElasticacheUserRotation

### Direktori Aktif

#### Kredensial Direktori Aktif

- Nama template: SecretsManagerActiveDirectoryRotationSingleUser
- **SecretString** Struktur yang diharapkan: [the section called “Struktur rahasia kredensi Direktori Aktif”](#).

- Kode sumber: [https://github.com/aws-samples/ aws-secrets-manager-rotation -lambdas/tree/ master/ /lambda\\_function.py](https://github.com/aws-samples/aws-secrets-manager-rotation-lambdas/tree/master/lambda_function.py) SecretsManagerActiveDirectoryRotationSingleUser

### Tab tombol Direktori Aktif

- Nama template: SecretsManagerActiveDirectoryAndKeytabRotationSingleUser
- **SecretString** Struktur yang diharapkan: [the section called “Struktur rahasia Direktori Aktif”](#).
- Kode sumber: [https://github.com/aws-samples/ aws-secrets-manager-rotation -lambdas/tree/ master/ /lambda\\_function.py](https://github.com/aws-samples/aws-secrets-manager-rotation-lambdas/tree/master/lambda_function.py) SecretsManagerActiveDirectoryAndKeytabRotationSingleUser
- 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/ /lambda\\_function.py](https://github.com/aws-samples/aws-secrets-manager-rotation-lambdas/tree/master/lambda_function.py) SecretsManagerRotationTemplate

## Izin peran eksekusi fungsi rotasi Lambda untuk AWS Secrets Manager

Karena [the section called “Rotasi oleh fungsi Lambda”](#), ketika Secrets Manager menggunakan fungsi Lambda untuk memutar rahasia, Lambda mengasumsikan [peran eksekusi IAM](#) dan memberikan kredensial 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](#).

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "secretsmanager:UpdateSecretVersionStage"
      ],
      "Resource": "SecretARN"
    },
    {
      "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, lihat [the 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 sarankan Anda melampirkan kebijakan AWS terkelola [ReadOnlyAccessAmazonRDS](#).

Contoh kebijakan berikut memungkinkan fungsi untuk:

- Jalankan operasi Secrets Manager untuk *SecretArn*.
- Ambil kredensialnya di rahasia superuser. Secrets Manager menggunakan kredensial dalam rahasia superuser untuk memperbarui kredensial 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](#).

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",

```

```

        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "secretsmanager:UpdateSecretVersionStage"
    ],
    "Resource": "SecretARN"
},
{
    "Effect": "Allow",
    "Action": [
        "secretsmanager:GetSecretValue"
    ],
    "Resource": "SuperuserSecretARN"
},
{
    "Effect": "Allow",
    "Action": [
        "secretsmanager:GetRandomPassword"
    ],
    "Resource": "*"
},
{
    "Action": [
        "ec2:CreateNetworkInterface",
        "ec2>DeleteNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DetachNetworkInterface"
    ],
    "Resource": "*",
    "Effect": "Allow"
}
]
}

```

## Akses jaringan untuk fungsi rotasi Lambda

Karena [the section called “Rotasi oleh 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 kredensial 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, lihat [the section called “Titik akhir Secrets Manager”](#).

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](#).

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.

## Memecahkan masalah rotasi AWS Secrets Manager

Untuk banyak layanan, Secrets Manager menggunakan fungsi Lambda untuk memutar rahasia. Untuk informasi selengkapnya, lihat [the section called “Rotasi oleh 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”](#).

Untuk melihat CloudWatch log untuk fungsi Lambda Anda

1. Buka konsol Secrets Manager di <https://console.aws.amazon.com/secretsmanager/>.
2. Pilih rahasia Anda, dan kemudian pada halaman detail, di bawah konfigurasi Rotasi, pilih fungsi rotasi Lambda. Konsol Lambda terbuka.
3. Pada tab Monitor, pilih Log, lalu pilih Lihat log masuk CloudWatch.

CloudWatch Konsol membuka dan menampilkan log untuk fungsi Anda.

Untuk menafsirkan log

- [Tidak ada aktivitas setelah “Menemukan kredensial 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 dari Python 3.7 ke 3.9](#)

Tidak ada aktivitas setelah “Menemukan kredensial dalam variabel lingkungan”

Jika tidak ada aktivitas setelah “Ditemukan kredensial 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, lihat [the section called “Titik akhir Secrets Manager”](#).

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 mencegah permintaan yang ditujukan ke titik akhir regional publik dan mengarahkannya ke titik akhir pribadi. Untuk informasi selengkapnya, lihat [Titik akhir VPC](#).

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 masuk ke 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 [Titik akhir VPC](#).

Rahasiannya 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 selengkapnya, 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 melihat `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, lihat [the 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, lihat [the 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 kredensial dalam rahasia. Periksa kredensial 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 `scram-sha-256` enkripsi

- Lihat Memeriksa pengguna dengan kata sandi non-Scram di blog [Otentikasi SCRAM di RDS](#) untuk PostgreSQL 13.

Untuk menentukan versi fungsi rotasi `libpq` Anda yang digunakan

1. Di komputer berbasis Linux, 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`:

```
0x0000000000000001d (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`:

```
0x0000000000000001d (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]
```

**Note**

Jika Anda mengatur rotasi rahasia otomatis sebelum 30 Desember 2021, fungsi rotasi Anda menggabungkan versi lama `libpq` yang tidak mendukung `scram-sha-256`. Untuk mendukung `scram-sha-256`, Anda perlu [membuat ulang fungsi rotasi Anda](#).

Database membutuhkan akses SSL/TLS.

Jika database Anda memerlukan koneksi SSL/TLS, 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 lama yang tidak mendukung SSL/TLS. Untuk mendukung koneksi yang menggunakan 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 3.9”](#) 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 dari Python 3.7 ke 3.9

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 3.9, 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 AWS Management Console 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 3.9:

- [Opsi 1: Buat ulang fungsi rotasi menggunakan AWS CloudFormation](#)
- [Opsi 2: Perbarui runtime untuk fungsi rotasi yang ada menggunakan AWS CloudFormation](#)
- [Opsi 3: Untuk AWS CDK pengguna, tingkatkan perpustakaan CDK](#)

## Opsi 1: Buat ulang fungsi rotasi menggunakan AWS CloudFormation

Saat Anda menggunakan konsol Secrets Manager untuk mengaktifkan rotasi, Secrets Manager menggunakan AWS CloudFormation untuk membuat sumber daya yang diperlukan, termasuk fungsi rotasi Lambda. Jika Anda menggunakan konsol untuk mengaktifkan rotasi, atau Anda membuat fungsi rotasi menggunakan AWS CloudFormation tumpukan, Anda dapat menggunakan AWS CloudFormation tumpukan yang sama untuk membuat ulang fungsi rotasi dengan nama baru. Fungsi baru menggunakan versi Python yang lebih baru.

Untuk menemukan AWS 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:us-west-2:408736277230:stack/SecretsManagerRDSMySQLRotationSingleUser5c2-SecretRotationScheduleHostedRotationLambda-3CUDHZMDMB08/79fc9050-2eef-11ed-`
- Nama tumpukan: **SecretsManagerRDSMySQLRotationSingleUser5c2-SecretRotationScheduleHostedRotationLambda**

Untuk membuat ulang fungsi rotasi ()AWS CloudFormation

1. Di AWS 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, pilih Edit template di desainer, lalu pilih View in Designer.
3. Dalam desainer, dalam kode template, di `SecretRotationScheduleHostedRotationLambda`, ganti nilai untuk `"functionName"`: `"SecretsManagerTestRotationRDS"` dengan nama fungsi baru, misalnya di JSON, **`"functionName": "SecretsManagerTestRotationRDSupdated"`**
4. Lanjutkan melalui alur kerja AWS CloudFormation tumpukan dan kemudian pilih Kirim.

## Opsi 2: Perbarui runtime untuk fungsi rotasi yang ada menggunakan AWS CloudFormation

Saat Anda menggunakan konsol Secrets Manager untuk mengaktifkan rotasi, Secrets Manager menggunakan AWS CloudFormation untuk membuat sumber daya yang diperlukan, termasuk fungsi rotasi Lambda. Jika Anda menggunakan konsol untuk mengaktifkan rotasi, atau Anda membuat fungsi rotasi menggunakan AWS CloudFormation tumpukan, Anda dapat menggunakan AWS CloudFormation tumpukan yang sama untuk memperbarui runtime untuk fungsi rotasi.

Untuk menemukan AWS 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:us-west-2:408736277230:stack/SecretsManagerRDSMySQLRotationSingleUser5c2-SecretRotationScheduleHostedRotationLambda-3CUDHZMDMB08/79fc9050-2eef-11ed-`
- Nama tumpukan: **SecretsManagerRDSMySQLRotationSingleUser5c2-SecretRotationScheduleHostedRotationLambda**

Untuk memperbarui runtime untuk fungsi rotasi ()AWS CloudFormation

1. Di AWS 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, pilih Edit template di desainer, lalu pilih View in Designer.
3. Di desainer, di template JSON, untuk, di bawah `SecretRotationScheduleHostedRotationLambda`, di bawah `PropertiesParameters`, tambahkan **"runtime": "python3.9"**
4. Lanjutkan melalui alur kerja AWS CloudFormation tumpukan dan kemudian pilih Kirim.

## Opsi 3: Untuk AWS CDK pengguna, tingkatkan perpustakaan CDK

Jika Anda menggunakan versi AWS CDK sebelumnya v2.94.0 untuk mengatur rotasi rahasia Anda, Anda dapat memperbarui fungsi Lambda dengan memutakhirkan ke v2.94.0 atau yang lebih baru.

Untuk informasi selengkapnya, lihat [Panduan Pengembang AWS Cloud Development Kit \(AWS CDK\) v2](#).

## Putar AWS Secrets Manager rahasia segera

Anda hanya dapat memutar rahasia yang telah dikonfigurasi rotasi. Untuk menentukan apakah rahasia telah dikonfigurasi untuk rotasi, di konsol, lihat rahasia dan gulir ke bawah ke bagian konfigurasi Rotasi. Jika status Rotasi Diaktifkan, maka rahasianya dikonfigurasi untuk rotasi. Jika belum, lihat [Putar rahasia](#).

Untuk segera memutar rahasia (konsol)

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 Rotate secret segera.
4. Dalam kotak dialog Rotate secret, pilih Rotate.

## AWS CLI

Example Putar rahasia segera

[rotate-secret](#) Contoh berikut memulai rotasi langsung. Rahasianya harus sudah memiliki rotasi yang dikonfigurasi.

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

## Jadwal rotasi

Saat Anda mengaktifkan rotasi otomatis, Anda dapat menggunakan ekspresi cron () atau rate () untuk mengatur jadwal untuk memutar rahasia Anda. Dengan ekspresi laju, Anda dapat membuat jadwal rotasi yang berulang pada interval jam atau hari. Dengan ekspresi cron, Anda dapat membuat jadwal rotasi yang lebih detail daripada interval rotasi. Jadwal rotasi Secrets Manager menggunakan zona waktu UTC. Anda dapat memutar rahasia sesering setiap empat jam. Secrets Manager memutar rahasia Anda kapan saja selama jendela rotasi.

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 \(konsol\)”](#)

## Ekspresi rate

Ekspresi tingkat Secrets Manager memiliki format berikut, di mana *Nilai* adalah bilangan bulat positif dan *Unit* dapat berupa `hour`, `hoursday`, atau `days`:

```
rate(Value Unit)
```

Anda dapat memutar rahasia sesering setiap empat jam. Contoh:

- `rate(4 hours)` berarti rahasia diputar setiap empat jam.
- `rate(1 day)` berarti rahasianya diputar setiap hari.
- `rate(10 days)` berarti rahasianya diputar setiap 10 hari.

Untuk tingkat dalam jam, jendela rotasi default dimulai pada tengah malam dan ditutup setelah satu jam. Anda dapat mengatur durasi Jendela untuk mengubah jendela rotasi. Jendela rotasi tidak boleh meluas ke jendela rotasi berikutnya. Salah satu cara untuk memeriksa ini adalah untuk mengkonfirmasi bahwa jendela rotasi kurang dari atau sama dengan jumlah jam antara rotasi.

Untuk tingkat dalam beberapa hari, jendela rotasi default dimulai pada tengah malam dan ditutup pada akhir hari. Anda dapat mengatur durasi Jendela untuk mengubah jendela rotasi. Jendela rotasi tidak boleh diperpanjang ke hari UTC berikutnya. Salah satu cara untuk memeriksa ini adalah dengan mengkonfirmasi bahwa jam mulai ditambah durasi jendela kurang dari atau sama dengan 24 jam.

## Ekspresi Cron

Ekspresi cron 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.

Untuk jadwal dalam jam, jendela rotasi default ditutup setelah satu jam. Anda dapat mengatur durasi Jendela untuk mengubah jendela rotasi. Jendela rotasi tidak boleh masuk ke jendela rotasi berikutnya. Anda dapat memutar rahasia sesering setiap empat jam.

Contoh jadwal	Ekspresi
Setiap delapan jam dimulai pada tengah malam.	<code>cron(0 /8 * * ? *)</code>
Setiap delapan jam mulai pukul 8:00 pagi.	<code>cron(0 8/8 * * ? *)</code>
Setiap sepuluh jam, mulai pukul 2:00 pagi.	<code>cron(0 2/10 * * ? *)</code>
Jendela rotasi akan dimulai pada 2:00, 12:00, dan 22:00, dan kemudian hari berikutnya pada 2:00, 12:00, dan 22:00.	
Setiap hari pukul 10:00 pagi.	<code>cron(0 10 * * ? *)</code>
Setiap hari Sabtu pukul 18.00.	<code>cron(0 18 ? * SAT *)</code>
Hari pertama setiap bulan pukul 8:00 pagi.	<code>cron(0 8 1 * ? *)</code>
Setiap tiga bulan pada hari Minggu pertama pukul 1:00 pagi.	<code>cron(0 1 ? 1/3 SUN#1 *)</code>
Hari terakhir setiap bulan pukul 17:00.	<code>cron(0 17 L * ? *)</code>
Senin sampai Jumat pukul 8:00 pagi.	<code>cron(0 8 ? * MON-FRI *)</code>
Hari pertama dan ke-15 setiap bulan pukul 16:00.	<code>cron(0 16 1,15 * ? *)</code>
Minggu pertama setiap bulan pada tengah malam.	<code>cron(0 0 ? * SUN#1 *)</code>

## 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 memutar rahasia sesering setiap empat jam.
Day-of-month	1–31	<p>Gunakan, (koma) untuk memasukkan nilai tambahan. Misalnya 1, 15 berarti hari pertama dan ke-15 setiap bulan.</p> <p>Gunakan - (tanda hubung) untuk menentukan rentang. Misalnya 1–15 berarti hari 1 sampai 15 dalam sebulan.</p> <p>Gunakan* (tanda bintang) untuk menyertakan semua nilai di bidang. Misalnya * berarti setiap hari dalam sebulan.</p> <p>Wildcard ? (tanda tanya) menentukan satu atau yang lain. Anda tidak dapat menentukan kolom Day-of-month dan Day-of-week</p>

Bidang	Nilai-nilai	Wildcard
		<p>dalam ekspresi cron yang sama. Jika Anda menentukan sebuah nilai di salah satu kolom, maka Anda harus menggunakan ? (tanda tanya) di kolom yang lain.</p> <p>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.</p> <p>Gunakan L untuk menentukan hari terakhir bulan itu.</p> <p>Gunakan <b>DAY</b>L untuk menentukan hari bernama terakhir dalam sebulan. Misalnya SUNL berarti hari Minggu terakhir setiap bulan.</p>

Bidang	Nilai-nilai	Wildcard
Bulan	1—12 atau JAN—DEC	<p>Gunakan, (koma) untuk memasukkan nilai tambahan. Misalnya, JAN, APR, JUL, OCT berarti Januari, April, Juli, dan Oktober.</p> <p>Gunakan - (tanda hubung) untuk menentukan rentang. Misalnya 1–3 berarti bulan 1 sampai 3 tahun.</p> <p>Gunakan* (tanda bintang) untuk menyertakan semua nilai di bidang. Misalnya * berarti setiap bulan.</p> <p>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.</p>

Bidang	Nilai-nilai	Wildcard
Day-of-week	1—7 atau MATAHARI-SAT	<p>Gunakan # untuk menentukan hari dalam seminggu dalam sebulan. Misalnya, TUE#3 berarti Selasa ketiga setiap bulan.</p> <p>Gunakan, (koma) untuk memasukkan nilai tambahan. Misalnya 1, 4 berarti hari pertama dan keempat dalam seminggu.</p> <p>Gunakan - (tanda hubung) untuk menentukan rentang. Misalnya 1-4 berarti hari 1 sampai 4 dalam seminggu.</p> <p>Gunakan* (tanda bintang) untuk menyertakan semua nilai di bidang. Misalnya * berarti setiap hari dalam seminggu.</p> <p>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.</p>

Bidang	Nilai-nilai	Wildcard
		Gunakan/(garis miring ke depan) untuk menentukan 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

## 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 data Multi-Wilayah Multi-Akun](#).

Untuk menilai apakah rahasia berputar

- 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, konfigurasi 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.

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 “AWS 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”](#)
- 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](#).

Untuk daftar lengkap layanan yang menggunakan rahasia, lihat [Layanan yang menggunakan rahasia](#).

# AWS Layanan 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 App2Container menggunakan 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 rahasia AWS Secrets Manager](#)
- [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](#)
- [Cara Amazon Mengelola Alur Kerja untuk Apache Airflow menggunakan AWS Secrets Manager](#)
- [AWS Marketplace](#)
- [Bagaimana AWS Migration Hub menggunakan AWS Secrets Manager](#)
- [Cara AWS Panorama menggunakan Secrets Manager](#)
- [Bagaimana AWS ParallelCluster menggunakan AWS Secrets Manager](#)
- [Bagaimana Amazon Q menggunakan Secrets Manager](#)
- [Bagaimana AWS OpsWorks for Chef Automate menggunakan AWS Secrets Manager](#)
- [Bagaimana Amazon QuickSight menggunakan AWS Secrets Manager](#)
- [Amazon RDS](#)
- [Bagaimana Amazon Redshift menggunakan AWS Secrets Manager](#)
- [Editor kueri Amazon Redshift v2](#)
- [Bagaimana Amazon SageMaker menggunakan AWS Secrets Manager](#)
- [Bagaimana AWS Schema Conversion Tool 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 rahasia AWS Secrets Manager](#)

## 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 App2Container menggunakan 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 kredensial 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 perintah CLI [rds execute-statement](#) untuk terhubung ke Amazon RDS menggunakan kredensial secara rahasia. 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 dikenakan biaya untuk rahasia itu. Aurora juga [mengelola rotasi untuk kredensial 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 [the section called “Buat rahasia database”](#)

Saat Anda memanggil Amazon RDS Data API, Anda dapat meneruskan kredensial 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 kredensial 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 kredensial 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 kredensial 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. DataSync Discovery membantu Anda mempercepat migrasi ke AWS.

Untuk mengumpulkan informasi tentang sistem penyimpanan lokal, DataSync Discovery menggunakan kredensial untuk antarmuka manajemen sistem penyimpanan. DataSync menyimpan kredensial tersebut dalam rahasia yang [dikelola](#) Secrets Manager dengan awalan. `datasync` Anda dikenakan biaya untuk rahasia itu. Untuk informasi selengkapnya, lihat [Menambahkan sistem penyimpanan lokal ke DataSync Discovery](#) di Panduan AWS DataSync 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 kluster 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

AWS Direct Connect menghubungkan jaringan internal Anda ke AWS Direct Connect lokasi melalui kabel serat optik Ethernet standar. Dengan koneksi ini, Anda dapat membuat antarmuka virtual langsung ke publik Layanan AWS.

AWS Direct Connect menyimpan nama kunci asosiasi konektivitas dan key pair asosiasi konektivitas (pasangan CKN/CAK) dalam [rahasia terkelola](#) dengan awalan. `directconnect` Biaya rahasia sudah termasuk dengan biaya untuk AWS Direct Connect. Untuk memperbarui rahasia, Anda harus menggunakan AWS Direct Connect bukan Secrets Manager. Untuk informasi selengkapnya, lihat [Mengaitkan CKN/CAK MacSec dengan LAG di Panduan Pengguna](#).AWS Direct Connect

## Bagaimana AWS Directory Service menggunakan AWS Secrets Manager

AWS 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 kredensial. Untuk informasi selengkapnya, di Panduan AWS 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

Di Amazon DocumentDB, pengguna mengautentikasi ke kluster bersama dengan kata sandi. Dengan AWS Secrets Manager, Anda dapat mengganti kredensial hard-code dalam kode Anda (termasuk kata sandi) dengan panggilan API ke Secrets Manager untuk mengambil rahasia secara programatis. Untuk informasi selengkapnya, lihat [the section called “Buat rahasia database”](#) dan [Mengelola Pengguna Amazon DocumentDB](#) 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 kredensial 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 kredensial 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 kredensi 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 untuk mengenkripsi/mendekripsi 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 kredensial 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 just-in-time video 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 Anda membuat Notebook EMR, Anda dapat menyimpan kredensial registri berbasis Git pribadi Anda menggunakan Secrets Manager. Untuk informasi selengkapnya, lihat [Menambahkan Repositori Berbasis Git 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 rahasia AWS Secrets Manager

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 kredensial 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 `dataabrew`. 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 dikenakan biaya 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 selengkapnya, 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 mengotentikasi. Untuk informasi selengkapnya, lihat [Mengonfigurasi otentikasi 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 kredensial 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 kredensial 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.

## Cara Amazon Mengelola Alur Kerja untuk Apache Airflow menggunakan AWS Secrets Manager

Amazon Managed Workflows for Apache Airflow adalah layanan orkestrasi terkelola untuk [Apache Airflow](#) yang memudahkan penyiapan dan pengoperasian end-to-end pipeline data 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 rahasia untuk variabel Apache Airflow di Amazon Managed Workflows AWS Secrets Manager 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](#).

## Cara AWS Panorama menggunakan Secrets Manager

AWS Panorama adalah layanan yang membawa visi komputer ke jaringan kamera lokal Anda. Anda gunakan AWS Panorama untuk mendaftarkan alat, memperbarui perangkat lunaknya, dan menyebarkan aplikasi ke dalamnya. Saat Anda mendaftarkan aliran video sebagai sumber data untuk aplikasi Anda, jika aliran dilindungi kata sandi, AWS Panorama menyimpan kredensialnya dalam rahasia Secrets Manager. Untuk informasi selengkapnya, lihat [Mengelola aliran kamera AWS Panorama di](#) Panduan AWS Panorama Pengembang.



## 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 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 kredensial 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 AWS OpsWorks for Chef Automate menggunakan AWS Secrets Manager

AWS 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 awalanopsworks-cm. Biaya rahasia sudah termasuk dalam biaya untuk AWS OpsWorks. Untuk informasi selengkapnya, lihat [Integrasi dengan AWS Secrets Manager](#) di Panduan AWS OpsWorks Pengguna.

## Bagaimana Amazon QuickSight menggunakan AWS Secrets Manager

Amazon QuickSight adalah layanan intelijen bisnis skala cloud (BI) yang dapat Anda gunakan untuk analitik, visualisasi data, dan pelaporan. Anda dapat menggunakan berbagai sumber data di Amazon QuickSight. Jika Anda menyimpan kredensial database dalam rahasia Secrets Manager, Amazon



QuickSight dapat menggunakan rahasia tersebut untuk terhubung ke database. Untuk informasi selengkapnya, lihat [Menggunakan AWS Secrets Manager rahasia sebagai pengganti kredensi database QuickSight di Amazon](#) di QuickSight Panduan Pengguna Amazon.

## Amazon RDS

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 dikenakan biaya untuk rahasia itu. Amazon RDS juga [mengelola rotasi](#) untuk kredensial 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. [the section called “Buat rahasia database”](#)

Saat Anda menggunakan editor kueri Amazon RDS untuk menyambung ke database, Anda dapat menyimpan kredensial 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 dikenakan biaya untuk rahasia itu. Amazon Redshift juga [mengelola rotasi untuk kredensial](#) ini. Untuk informasi selengkapnya, lihat [Mengelola kata sandi admin Amazon Redshift menggunakan AWS Secrets Manager](#) dalam Panduan Manajemen Amazon Redshift.

Untuk kredensial Amazon Redshift lainnya, lihat. [the section called “Buat rahasia database”](#)

Saat memanggil Amazon Redshift Data API, Anda dapat meneruskan kredensial untuk kluster 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 kredensial Anda dalam rahasia Secrets Manager dengan awalan.

`redshiftqueryeditor` Anda dikenakan biaya untuk rahasia itu. Untuk informasi selengkapnya, lihat [Menanyakan database menggunakan editor kueri](#) di Panduan Manajemen Amazon Redshift.

Untuk editor kueri v2, lihat [the 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 kredensial 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, lihat [the section called “Amazon Redshift”](#).

## Bagaimana Amazon SageMaker menggunakan AWS Secrets Manager

SageMaker adalah layanan pembelajaran mesin yang dikelola sepenuhnya. Dengan SageMaker, 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 .

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 kredensial Anda 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 kredensial 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 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 kluster 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 memberikan 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 rahasia AWS Secrets Manager

AWS Wickr adalah layanan end-to-end terenkripsi yang membantu organisasi dan lembaga pemerintah untuk berkomunikasi dengan aman melalui one-to-one dan mengelompokkan pesan, 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 Administrasi AWS Wickr.

# 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 Anda 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. AWS 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 nama layanan: `com.amazonaws.wilayah.secretsmanager`
2. Untuk mengontrol akses ke titik akhir, lihat [Mengontrol akses ke titik akhir VPC menggunakan kebijakan titik akhir](#).

## 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.

# 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 pada [Buat 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 AWS CloudFormation, lihat [Mempelajari dasar-dasar templat](#) di Panduan AWS 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 AWS 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 AWS CloudFormation, lihat [Mempelajari dasar-dasar templat](#) di Panduan AWS CloudFormation Pengguna.

## JSON

```
{
  "Resources": {
    "CloudFormationCreatedSecret": {
      "Type": "AWS::SecretsManager::Secret",
      "Properties": {
        "Description": "Simple secret created by AWS CloudFormation.",
        "GenerateSecretString": {
          "SecretStringTemplate": "{\"username\": \"saanvi\"}",
          "GenerateStringKey": "password",
          "PasswordLength": 32
        }
      }
    }
  }
}
```

## YAML

```
Resources:
  CloudFormationCreatedSecret:
    Type: 'AWS::SecretsManager::Secret'
    Properties:
      Description: Simple secret created by AWS CloudFormation.
      GenerateSecretString:
        SecretStringTemplate: '{"username": "saanvi"}'
        GenerateStringKey: password
```



```
PasswordLength: 32
```

## Buat AWS Secrets Manager rahasia dengan rotasi otomatis dan instans Amazon RDS MySQL DB dengan AWS 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. Untuk informasi selengkapnya, lihat [Rotasi terkelola](#).

## Buat AWS Secrets Manager rahasia dan cluster Amazon Redshift dengan AWS 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 AWS CloudFormation

Contoh ini membuat rahasia dan instance Amazon DocumentDB menggunakan kredensi dalam rahasia sebagai pengguna dan kata sandi. Rahasiannya 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 AWS CloudFormation, lihat [Mempelajari dasar-dasar templat](#) di Panduan AWS 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":[]
    }
  }
}
```



```

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, AWS CloudFormation buat [AWS::Serverless::Function](#) berdasarkan yang sesuai [Templat fungsi rotasi](#). Kemudian AWS CloudFormation atur [RotationSchedule](#), yang mengatur fungsi rotasi dan aturan rotasi untuk rahasia. Anda dapat melihat AWS CloudFormation tumpukan dengan memilih View stack di banner setelah Anda mengaktifkan rotasi otomatis.

Untuk informasi tentang mengaktifkan rotasi otomatis, lihat [Putar 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](#).

# 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 peristiwa dengan AWS CloudTrail](#)
- [Monitor AWS Secrets Manager dengan Amazon CloudWatch](#)
- [Cocokkan 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](#)

## Log AWS Secrets Manager peristiwa 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 dalam catatan Secrets Manager, lihat [CloudTrail entri](#).

Anda dapat menggunakan CloudTrail konsol untuk melihat 90 hari terakhir dari peristiwa yang direkam. Untuk catatan peristiwa yang sedang berlangsung di AWS akun Anda, termasuk peristiwa untuk Secrets Manager, 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 menindaklanjuti 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 [Mengonfigurasi notifikasi Amazon SNS](#) untuk CloudTrail

Untuk mengambil peristiwa Secrets Manager 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 kriteria Filter dan/atau rentang waktu untuk membantu Anda menemukan acara yang Anda cari. Sebagai contoh:
  - a. Untuk melihat semua event Secrets Manager, untuk atribut Lookup, pilih Event source. Kemudian, untuk Masukkan sumber acara, pilih **secretsmanager.amazonaws.com**.
  - b. Untuk melihat semua peristiwa untuk rahasia, untuk atribut Pencarian, pilih Nama sumber daya. Kemudian, untuk Masukkan nama sumber daya, masukkan nama rahasianya.
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 Secrets Manager dari log CloudTrail

[lookup-events](#) Contoh berikut mencari peristiwa Secrets Manager.

```
aws cloudtrail lookup-events \  
  --region us-east-1 \  
  --lookup-attributes  
  AttributeKey=EventSource,AttributeValue=secretsmanager.amazonaws.com
```

## AWS CloudTrail entri untuk Secrets Manager

AWS Secrets Manager menulis entri ke AWS CloudTrail log Anda untuk semua operasi Secrets Manager dan untuk acara lain yang terkait dengan rotasi dan penghapusan. Untuk informasi tentang mengambil tindakan pada peristiwa ini, lihat [Acara 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 Secrets Manager memiliki "detail-type": ["AWS API Call via CloudTrail"].

### Note

Sebelum Februari 2024, beberapa operasi Secrets Manager melaporkan peristiwa yang berisi "aRn" bukan "arn" untuk ARN rahasia. Untuk informasi lebih lanjut, lihat [AWS re:Post](#).

Berikut ini adalah CloudTrail entri yang dihasilkan saat Anda atau layanan memanggil Secrets Manager beroperasi melalui API, SDK, atau CLI.

### BatchGetSecretValue

Dihasilkan oleh [BatchGetSecretValue](#) operasi. Untuk informasi tentang mengambil rahasia, lihat [Dapatkan rahasia](#).

### CancelRotateSecret

Dihasilkan oleh [CancelRotateSecret](#) operasi. Untuk informasi tentang rotasi, lihat [Putar rahasia](#).

### CreateSecret

Dihasilkan oleh [CreateSecret](#) operasi. Untuk informasi tentang membuat rahasia, lihat [Buat dan kelola rahasia](#).

### DeleteResourcePolicy

Dihasilkan oleh [DeleteResourcePolicy](#) operasi. Untuk informasi tentang izin, lihat [Kontrol autentikasi dan akses](#).

### DeleteSecret

Dihasilkan oleh [DeleteSecret](#) operasi. Untuk informasi tentang menghapus rahasia, lihat [the section called "Hapus rahasia"](#).

### DescribeSecret

Dihasilkan oleh [DescribeSecret](#) operasi.

### GetRandomPassword

Dihasilkan oleh [GetRandomPassword](#) operasi.

## GetResourcePolicy

Dihasilkan oleh [GetResourcePolicy](#) operasi. Untuk informasi tentang izin, lihat [Kontrol autentikasi dan akses](#).

## GetSecretValue

Dihasilkan oleh [GetSecretValue](#) dan [BatchGetSecretValue](#) operasi. Untuk informasi tentang mengambil rahasia, lihat [Dapatkan rahasia](#).

## ListSecrets

Dihasilkan oleh [ListSecrets](#) operasi. Untuk informasi tentang daftar rahasia, lihat [the section called "Temukan rahasia"](#).

## ListSecretVersionIds

Dihasilkan oleh [ListSecretVersionIds](#) operasi.

## PutResourcePolicy

Dihasilkan oleh [PutResourcePolicy](#) operasi. Untuk informasi tentang izin, lihat [Kontrol autentikasi dan akses](#).

## PutSecretValue

Dihasilkan oleh [PutSecretValue](#) operasi. Untuk informasi tentang memperbarui rahasia, lihat [the section called "Memodifikasi rahasia"](#).

## RemoveRegionsFromReplication

Dihasilkan oleh [RemoveRegionsFromReplication](#) operasi. Untuk informasi tentang mereplikasi rahasia, lihat [Replikasi rahasia di seluruh Wilayah](#).

## ReplicateSecretToRegions

Dihasilkan oleh [ReplicateSecretToRegions](#) operasi. Untuk informasi tentang mereplikasi rahasia, lihat [Replikasi rahasia di seluruh Wilayah](#).

## RestoreSecret

Dihasilkan oleh [RestoreSecret](#) operasi. Untuk informasi tentang memulihkan rahasia yang dihapus, lihat [the section called "Kembalikan rahasia"](#).

## RotateSecret

Dihasilkan oleh [RotateSecret](#) operasi. Untuk informasi tentang rotasi, lihat [Putar rahasia](#).

## StopReplicationToReplica

Dihasilkan oleh [StopReplicationToReplica](#) operasi. Untuk informasi tentang mereplikasi rahasia, lihat [Replikasi rahasia di seluruh Wilayah](#).

## TagResource

Dihasilkan oleh [TagResource](#) operasi. Untuk informasi tentang menandai rahasia, lihat [the section called "Rahasia tag"](#).

## UntagResource

Dihasilkan oleh [UntagResource](#) operasi. Untuk informasi tentang membuka tanda rahasia, lihat [the section called "Rahasia tag"](#).

## UpdateSecret

Dihasilkan oleh [UpdateSecret](#) operasi. Untuk informasi tentang memperbarui rahasia, lihat [the section called "Memodifikasi rahasia"](#).

## UpdateSecretVersionStage

Dihasilkan oleh [UpdateSecretVersionStage](#) operasi. Untuk informasi tentang tahapan versi, lihat [the section called "Versi rahasia"](#).

## ValidateResourcePolicy

Dihasilkan oleh [ValidateResourcePolicy](#) operasi. Untuk informasi tentang izin, lihat [Kontrol autentikasi dan akses](#).

## Entri log untuk penghapusan

Selain acara untuk operasi Secrets Manager, Secrets Manager menghasilkan peristiwa berikut yang terkait dengan penghapusan. Peristiwa ini memiliki "detail-type": ["AWS Service Event via CloudTrail"].

## CancelSecretVersionDelete

Dihasilkan oleh layanan Secrets Manager. Jika Anda memanggil `DeleteSecret` rahasia yang memiliki versi, dan kemudian menelepon `RestoreSecret`, Secrets Manager mencatat peristiwa ini untuk setiap versi rahasia yang dipulihkan. Untuk informasi tentang memulihkan rahasia yang dihapus, lihat [the section called "Kembalikan rahasia"](#).

## EndSecretVersionDelete

Dihasilkan oleh layanan Secrets Manager ketika versi rahasia dihapus. Untuk informasi selengkapnya, lihat [the section called "Hapus rahasia"](#).

## StartSecretVersionDelete

Dihasilkan oleh layanan Secrets Manager saat Secrets Manager memulai penghapusan untuk versi rahasia. Untuk informasi tentang menghapus rahasia, lihat [the section called "Hapus rahasia"](#).

## SecretVersionDeletion

Dihasilkan oleh layanan Secrets Manager saat Secrets Manager menghapus versi rahasia yang tidak digunakan lagi. Untuk informasi selengkapnya, lihat [Versi rahasia](#).

## Entri log untuk replikasi

Selain acara untuk operasi Secrets Manager, Secrets Manager menghasilkan peristiwa berikut yang terkait dengan replikasi. Peristiwa ini memiliki "detail-type": ["AWS Service Event via CloudTrail"].

### ReplicationFailed

Dihasilkan oleh layanan Secrets Manager saat replikasi gagal. Untuk informasi tentang mereplikasi rahasia, lihat [Replikasi rahasia di seluruh Wilayah](#).

### ReplicationStarted

Dihasilkan oleh layanan Secrets Manager saat Secrets Manager mulai mereplikasi rahasia. Untuk informasi tentang mereplikasi rahasia, lihat [Replikasi rahasia di seluruh Wilayah](#).

### ReplicationSucceeded

Dihasilkan oleh layanan Secrets Manager ketika sebuah rahasia berhasil direplikasi. Untuk informasi tentang mereplikasi rahasia, lihat [Replikasi rahasia di seluruh Wilayah](#).

## Entri log untuk rotasi

Selain acara untuk operasi Secrets Manager, Secrets Manager menghasilkan peristiwa berikut yang terkait dengan rotasi. Peristiwa ini memiliki "detail-type": ["AWS Service Event via CloudTrail"].

## RotationStarted

Dihasilkan oleh layanan Secrets Manager saat Secrets Manager mulai memutar rahasia. Untuk informasi tentang rotasi, lihat [Putar rahasia](#).

## RotationAbandoned

Dihasilkan oleh layanan Secrets Manager ketika Secrets Manager meninggalkan upaya rotasi dan menghapus AWSPENDING label dari versi rahasia yang ada. Secrets Manager meninggalkan rotasi saat Anda membuat versi baru rahasia selama rotasi. Untuk informasi tentang rotasi, lihat [Putar rahasia](#).

## RotationFailed

Dihasilkan oleh layanan Secrets Manager saat rotasi gagal. Untuk informasi tentang rotasi, lihat [the section called “Memecahkan masalah rotasi”](#).

## RotationSucceeded

Dihasilkan oleh layanan Secrets Manager ketika sebuah rahasia berhasil diputar. Untuk informasi tentang rotasi, lihat [Putar rahasia](#).

## TestRotationStarted

Dihasilkan oleh layanan Secrets Manager saat Secrets Manager mulai menguji rotasi untuk rahasia yang tidak dijadwalkan untuk rotasi langsung. Untuk informasi tentang rotasi, lihat [Putar rahasia](#).

## TestRotationSucceeded

Dihasilkan oleh layanan Secrets Manager ketika Secrets Manager berhasil menguji rotasi untuk rahasia yang tidak dijadwalkan untuk rotasi langsung. Untuk informasi tentang rotasi, lihat [Putar rahasia](#).

## TestRotationFailed

Dihasilkan oleh layanan Secrets Manager ketika Secrets Manager menguji rotasi untuk rahasia yang tidak dijadwalkan untuk rotasi langsung dan rotasi gagal. Untuk informasi tentang rotasi, lihat [the 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 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 `SecretsManagerResourceCount`, 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](#).

## Cocokkan 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, lihat [CloudTrail entri](#). Untuk petunjuk penyiapan EventBridge, lihat [Memulai EventBridge](#) di Panduan EventBridge Pengguna.

### 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 `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

Contoh berikut menunjukkan pola EventBridge peristiwa yang cocok dengan entri CloudTrail log untuk perubahan nilai rahasia yang terjadi dari pembaruan manual atau rotasi otomatis. Karena beberapa peristiwa ini berasal dari operasi Secrets Manager dan beberapa dihasilkan oleh layanan Secrets Manager, Anda harus menyertakan `detail-type` untuk keduanya.

```
{  
  "source": ["aws.secretsmanager"],  
  "$or": [  
    { "detail-type": ["AWS API Call via CloudTrail"] },  
    { "detail-type": ["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: Konfigurasi 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 menyetik 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 konfigurasi 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 data Multi-Wilayah Multi-Akun](#).

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”](#) dan [the section called “Log dengan AWS CloudTrail”](#).

Untuk informasi tentang AWS Secrets Manager harga, lihat [the section called “Harga”](#).

# 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 Whitepaper Keamanan dan Kepatuhan HIPAA — Whitepaper](#) ini menjelaskan bagaimana perusahaan dapat menggunakan untuk membuat aplikasi yang sesuai dengan HIPAA. AWS
- [AWS Sumber Daya AWS](#) — Kumpulan buku kerja dan panduan ini mungkin berlaku untuk industri dan lokasi Anda.
- AWS Config menilai 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](#) memberikan 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 penggunaan Security Hub guna mengevaluasi sumber daya Secrets Manager, lihat [AWS Secrets Manager kontrol](#) di Panduan AWS Security Hub 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 Manager menyediakan 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 AWS Secrets Manager sebagai layanan yang memenuhi syarat HIPAA.](#) Jika Anda memiliki Business Associate Agreement (BAA) yang dieksekusi AWS, Anda dapat menggunakan Secrets Manager untuk membantu membangun aplikasi yang sesuai dengan HIPAA Anda. AWS menawarkan [whitepaper yang berfokus pada HIPAA](#) untuk pelanggan yang tertarik untuk mempelajari lebih lanjut tentang bagaimana mereka dapat memanfaatkan pemrosesan dan AWS penyimpanan informasi kesehatan. Untuk informasi lebih lanjut, lihat [Kepatuhan HIPAA](#).
- **PIC Participating Organization** — 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 untuk Timur/Barat GovCloud dan untuk mengkonsumsi data pemerintah atau yang 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 Assessors 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. Auditor pihak ketiga secara teratur menguji dan memverifikasi keefektifan keamanan kami sebagai bagian dari [program kepatuhan AWS](#). Untuk mempelajari program kepatuhan yang berlaku di AWS Secrets Manager, lihat [Cakupan Layanan Menurut Program Kepatuhan AWS](#).
- Keamanan di cloud — AWS Layanan Anda menentukan tanggung jawab Anda. Anda juga bertanggung jawab atas faktor lain termasuk sensitivitas data Anda, persyaratan perusahaan Anda, serta hukum dan peraturan yang berlaku.

Untuk sumber daya lainnya, lihat [Pilar Keamanan - Kerangka AWS Well-Architected](#).

## Topik

- [Mengurangi risiko menggunakan AWS CLI untuk menyimpan rahasia Anda AWS Secrets Manager](#)
- [Perlindungan data di AWS Secrets Manager](#)
- [Enkripsi rahasia dan dekripsi di AWS Secrets Manager](#)
- [Keamanan infrastruktur dalam AWS Secrets Manager](#)
- [Ketahanan di AWS Secrets Manager](#)
- [TLS pasca-kuantum](#)

## 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 mengkompromikan 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 yang khas 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, lalu pilih Properties.
3. Pada tab Options, atur Buffer Size dan Number of 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 yang 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.
```

# Perlindungan data di AWS Secrets Manager

[Model tanggung jawab bersama](#) AWS diterapkan untuk perlindungan data AWS Secrets Manager. Sebagaimana dijelaskan dalam model ini, AWS 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. Untuk informasi lebih lanjut tentang privasi data, lihat [FAQ tentang Privasi Data](#). Untuk informasi tentang perlindungan data di Eropa, lihat postingan blog [Model Tanggung Jawab Bersama AWS dan GDPR](#) di Blog Keamanan AWS.

Untuk tujuan perlindungan data, sebaiknya Anda melindungi kredensial Akun AWS dan menyiapkan akun pengguna individu dengan AWS Identity and Access Management (IAM). Dengan cara seperti itu, 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 melakukan komunikasi dengan sumber daya AWS. 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 log aktivitas pengguna dengan AWS CloudTrail. Lihat [the section called “Log dengan AWS CloudTrail”](#).
- Jika Anda memerlukan modul kriptografi tervalidasi FIPS 140-2 ketika mengakses AWS melalui antarmuka baris perintah atau API, gunakan titik akhir FIPS. Lihat [the section called “Titik akhir Secrets Manager”](#).
- 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 tidak aktif

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 dalam transit

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 sertifikat X.509 dan/atau 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”](#).

## Privasi lalu lintas antar jaringan

AWS menawarkan opsi untuk menjaga privasi saat merutekan lalu lintas melalui rute jaringan yang dikenal dan pribadi.

Lalu lintas antara layanan dan aplikasi dan klien lokal

Anda memiliki dua opsi konektivitas antara jaringan privat dan AWS Secrets Manager:

- Koneksi Site-to-Site VPN AWS. Untuk informasi selengkapnya, lihat [Apa itu AWS VPN Site-to-Site?](#)
- Koneksi AWS Direct Connect. Untuk informasi selengkapnya, lihat [Apa itu AWS Direct Connect?](#)

Lalu lintas antara sumber daya AWS di Wilayah yang sama

Jika Anda ingin mengamankan lalu lintas antara Secrets Manager dan klien API AWS, siapkan [AWS PrivateLink](#) untuk 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. Untuk informasi selengkapnya, lihat [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, lihat [the section called “Ubah kunci enkripsi untuk rahasia”](#).

Saat Anda mengubah kunci enkripsi, Secrets Manager mengenkripsi ulang AWSCURRENT, AWSPENDING, 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.

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 [ListSecrets](#) atau [DescribeSecret](#). Ketika rahasia dikaitkan dengan Kunci yang dikelola AWS for Secrets Manager (`aws/secretsmanager`), operasi ini tidak mengembalikan pengenalan 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 [GenerateDataKey](#) operasi 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. Perbanyak hibah kunci KMS dapat memakan waktu hingga lima menit. Untuk informasi lebih lanjut, 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 [GenerateDataKey](#) operasi sebagai tanggapan atas operasi Secrets Manager berikut.

- [CreateSecret](#)— Jika rahasia baru menyertakan nilai rahasia, Secrets Manager meminta kunci data baru untuk mengenkripsi itu.
- [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.

[RotateSecret](#) Operasi 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.

- [GetSecretValue](#) dan [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`
- [PutSecretValue](#) dan [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.
- [ReplicateSecretToRegions](#)— Secrets Manager mengenkripsi ulang kunci data selama replikasi menggunakan kunci KMS di Region replika.

## DescribeKey

Secrets Manager memanggil [DescribeKey](#) operasi 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.

```
{
  "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 seperangkat pasangan nilai kunci yang berisi data non-rahasia yang berubah-ubah. 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 [GenerateDataKey](#) dan [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 CloudTrail](#) dan 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 [GenerateDataKey](#) dan [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, lihat [Memantau rahasia](#).

### GenerateDataKey

Saat Anda membuat atau mengubah nilai rahasia dalam rahasia, Secrets Manager mengirimkan [GenerateDataKey](#) permintaan 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": "AROAIQDTESTANDEXAMPLE: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, 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 KMS memperoleh ID kunci KMS dari ciphertext.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AROAIIGDTESTANDEXAMPLE: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": "AROAIQDTESTANDEXAMPLE: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 dalam 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. Untuk informasi selengkapnya, lihat [Titik akhir VPC](#).

## 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 [Reliability Pillar - AWS Well-Architected Framework](#).

Untuk informasi selengkapnya tentang Wilayah AWS dan Availability Zone, lihat [Infrastruktur AWS Global](#).

## TLS pasca-kuantum

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.

Guna melindungi data yang dienkripsi saat ini terhadap potensi serangan masa depan, AWS berpartisipasi dengan komunitas kriptografi dalam pengembangan algoritme 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 China.

Untuk mengkonfigurasi TLS pasca-kuantum hibrida

1. Tambahkan klien Common Runtime AWS untuk 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 inialisasi. 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](#).
- [Memperkenalkan s2n-tls, Implementasi dan Penggunaan s2n-tls TLS Open Source Baru](#).
- [Kriptografi Pasca-Kuantum](#) di Institut Nasional untuk Standar dan Teknologi (NIST).

- [Metode Enkapsulasi Kunci Pasca-Quantum Hybrid \(PQ KEM\) untuk Transport Layer Security 1.2 \(TLS\)](#).

TLS pasca-kuantum untuk 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, lihat [the section called “Memecahkan masalah rotasi”](#).

## Topik

- [Pesan “Akses ditolak”](#)
- [“Akses ditolak” untuk kredensial keamanan sementara](#)
- [Perubahan yang saya buat tidak selalu langsung terlihat.](#)
- [“Tidak dapat menghasilkan kunci data dengan kunci KMS asimetris” saat membuat rahasia](#)
- [Operasi AWS CLI atau AWS SDK tidak dapat menemukan rahasia saya dari ARN sebagian](#)
- [Rahasia ini dikelola oleh AWS layanan, dan Anda harus menggunakan layanan itu untuk memperbaruinya.](#)

## Pesan “Akses ditolak”

Saat melakukan panggilan API seperti `GetSecretValue` atau `CreateSecret` ke Secrets Manager, Anda harus memiliki izin IAM untuk melakukan panggilan tersebut. 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 mencakup kondisi apa pun, seperti `time-of-day` atau pembatasan alamat IP, Anda juga harus memenuhi persyaratan tersebut saat 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, lihat [Kontrol autentikasi dan akses](#).

Jika Anda menandatangani permintaan API secara manual, tanpa menggunakan [AWS SDK](#), verifikasi bahwa Anda [menandatangani permintaan](#) dengan benar.

## “Akses ditolak” untuk kredensial keamanan sementara

Verifikasi pengguna IAM atau peran yang Anda gunakan untuk membuat permintaan memiliki izin yang benar. Izin untuk kredensial 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 bagaimana 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, lihat [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

## “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](#).

## Operasi AWS CLI atau AWS 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.

Detail lebih lanjut

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 menemukannya.

```
$ aws secretsmanager describe-secret --secret-id arn:aws:secretsmanager:us-east-2:111122223333:secret:MySecret-abcdef
```

Rahasia ini dikelola oleh 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 yang dikelola](#).

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`

## Kuota AWS Secrets Manager

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 Disesan	Deskripsi
Tingkat gabungan permintaan <code>DeleteResourcePolicy</code> , <code>GetResourcePolicy</code> , <code>PutResourcePolicy</code> , dan <code>ValidateResourcePolicy</code> API	Setiap Wilayah yang didukung: 50 per detik	Tidak	Transaksi maksimum per detik untuk <code>DeleteResourcePolicy</code> , <code>GetResourcePolicy</code> , <code>PutResourcePolicy</code> , dan permintaan <code>ValidateResourcePolicy</code> API digabungkan.

Nama	Default	Dapat Disesan	Deskripsi
Tingkat gabungan permintaan DescribeSecret dan GetSecretValue API	Setiap Wilayah yang didukung: 10.000 per detik	Tidak	Transaksi maksimum per detik untuk DescribeSecret dan permintaan GetSecretValue API digabungkan.
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.

Nama	Default	Dapat Dيسان	Deskripsi
Tingkat permintaan BatchGetSecretValue API	Setiap Wilayah yang didukung: 100 per detik	Tidak	Transaksi maksimum per detik untuk permintaan BatchGetSecretValue API.
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 GetRandom Password API	Setiap Wilayah yang didukung: 50 per detik	Tidak	Transaksi maksimum per detik untuk permintaan GetRandomPassword API.
Tingkat permintaan ListSecretVersionIds 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.
Panjang kebijakan berbasis sumber daya	Setiap Wilayah yang didukung: 20.480	Tidak	Jumlah maksimum karakter dalam kebijakan izin berbasis sumber daya yang dilampirkan pada rahasia.

Nama	Default	Dapat Disesuaikan	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

AWSKlien Anda mungkin melihat panggilan ke Secrets Manager gagal karena masalah tak terduga di sisi klien. 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

## Pelambatan sisi layanan dan kesalahan dan pengecualian batas

- Throttling
- ThrottlingException
- ThrottledException
- RequestThrottledException
- TooManyRequestsException
- ProvisionedThroughputExceededException
- TransactionInProgressException
- RequestLimitExceeded
- BandwidthLimitExceeded
- LimitExceededException
- RequestThrottled
- SlowDown

Untuk informasi lebih lanjut, serta kode contoh, pada percobaan ulang, backoff eksponensial, dan jitter, lihat sumber daya berikut:

- [Backoff dan Jitter Eksponensial](#)
- [Batas waktu, percobaan ulang, dan backoff dengan jitter](#)
- [Error pengulangan dan mundur eksponensial dalam 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
<a href="#">Secrets Manager berubah menjadi kebijakan AWS terkelola</a>	Kebijakan SecretsManagerReadWrite terkelola sekarang termasuk redshift-serverless izin. Untuk informasi selengkapnya, lihat <a href="#">kebijakan AWS terkelola AWS Secrets Manager</a>	Maret 12, 2024

## Pembaruan sebelumnya

Tabel berikut menjelaskan perubahan penting dalam setiap rilis Panduan AWS Secrets Manager Pengguna sebelum Februari 2024.

Perubahan	Deskripsi	Tanggal
Ketersediaan umum	Ini adalah rilis publik perdana Secrets Manager.	Apr 4, 2018

Terjemahan disediakan oleh mesin penerjemah. Jika konten terjemahan yang diberikan bertentangan dengan versi bahasa Inggris aslinya, utamakan versi bahasa Inggris.