Guida per l'utente

Gestione dei segreti AWS

Gestione dei segreti AWS: Guida per l’utente

Table of Contents

Che cos'è Gestione dei segreti AWS?

Gestione dei segreti AWS ti aiuta a gestire, recuperare e ruotare le credenziali del database, le credenziali delle applicazioni, i OAuth token, le chiavi API e altri segreti durante il loro ciclo di vita. Molti AWS servizi archiviano e utilizzano segreti in Secrets Manager.

Secrets Manager consente di migliorare l'assetto di sicurezza, perché non sono più necessarie credenziali a codifica fissa nel codice sorgente dell'applicazione. L'archiviazione delle credenziali in Secrets Manager aiuta a evitare possibili compromissioni da parte di chiunque che possa esaminare la tua applicazione o i tuoi componenti. Puoi sostituire le credenziali a codifica fissa con una chiamata a runtime al servizio Secrets Manager per recuperare le credenziali in modo dinamico quando ne hai bisogno.

Con Secrets Manager è possibile impostare un programma automatico di rotazione automatica per i segreti. In questo modo puoi sostituire i segreti a lungo termine con altri a breve termine, riducendo notevolmente il rischio di compromissione. Poiché le credenziali non sono più archiviate con l'applicazione, la rotazione delle credenziali non richiede più l'aggiornamento delle applicazioni e l'implementazione di modifiche ai client delle applicazioni.

Per altri segreti che potresti avere nell'organizzazione:

- AWS credenziali: consigliamo AWS Identity and Access Management.
- Chiavi di crittografia: consigliamo AWS Key Management Service.
- Chiavi SSH: consigliamo Amazon EC2 Instance Connect.
- Chiavi e certificati privati: consigliamo AWS Certificate Manager.

Inizia a usare Secrets Manager

Se non conosci Secrets Manager, inizia con uno dei seguenti tutorial:

- the section called “Sostituzione dei segreti codificati ”
- the section called “Sostituzione delle credenziali del database codificato ”
- the section called “Rotazione a utenti alternati”
- the section called “Rotazione a utente singolo”

Altre attività che puoi eseguire con i segreti:

- Gestisci i segreti
- Controllare l'accesso ai tuoi segreti
- Ottieni segreti
- Rotazione dei segreti
- Monitorare i segreti
- Monitora i segreti ai fini della conformità
- Crea segreti in AWS CloudFormation

Conformità agli standard

Gestione dei segreti AWS è stato sottoposto a controlli per i diversi standard e può far parte della vostra soluzione quando è necessario ottenere la certificazione di conformità. Per ulteriori informazioni, consulta Convalida della conformità.

Prezzi

Quando usi Secrets Manager paghi soltanto per ciò che utilizzi, senza tariffe minime o per la configurazione. Non è previsto alcun costo per i segreti che sono contrassegnati per l'eliminazione. Per l'elenco completo dei prezzi aggiornati, consulta la pagina dei prezzi Gestione dei segreti AWS. Per monitorare i costi, consulta. the section called "Monitora i costi di Secrets Manager"

Puoi utilizzare il Chiave gestita da AWS `aws/secretsmanager` programma creato da Secrets Manager per crittografare i tuoi segreti gratuitamente. Se crei le tue chiavi KMS per crittografare i tuoi segreti, ti AWS addebiterà la tariffa attuale. AWS KMS Per ulteriori informazioni, consultare AWS Key Management Service Prezzi.

Quando attivi la rotazione automatica (eccetto la rotazione gestita), Secrets Manager utilizza una AWS Lambda funzione per ruotare il segreto e ti viene addebitato il costo della funzione di rotazione alla velocità Lambda corrente. Per ulteriori informazioni, consultare AWS Lambda Prezzi.

Se abiliti AWS CloudTrail il tuo account, puoi ottenere i log delle chiamate API inviate da Secrets Manager. Secrets Manager registra tutti gli eventi come eventi di gestione. AWS CloudTrail archivia gratuitamente la prima copia di tutti gli eventi di gestione. Tuttavia, ti potrebbero venire addebitati dei costi per lo storage dei log per Amazon S3 e Amazon SNS se abiliti la notifica. Inoltre, se imposti ulteriori trail, le copie aggiuntive di eventi di gestione potrebbero comportare dei costi. Per ulteriori informazioni, consultare Prezzi di AWS CloudTrail.

È possibile utilizzare i tag di allocazione dei costi in Secrets Manager per tenere traccia e classificare le spese associate a segreti o progetti specifici. Per ulteriori informazioni, consulta the section called "Tag segreti " questa guida e Utilizzo AWS dei tag di allocazione dei costi nella Guida per l' AWS Billing utente.

Accesso Gestione dei segreti AWS

Puoi usare il servizio Secrets Manager in uno dei modi seguenti:

- Console Secrets Manager
- Strumenti a riga di comando
- AWS SDKs
- API query HTTPS
- Gestione dei segreti AWS endpoint

Console Secrets Manager

Puoi gestire i segreti utilizzando la console Secrets Manager basata su browser ed eseguire quasi tutte le attività correlate ai segreti utilizzando la console.

Strumenti a riga di comando

Gli strumenti da riga di AWS comando consentono di impartire comandi dalla riga di comando del sistema per eseguire Secrets Manager e altre AWS attività. Questa modalità può risultare più veloce e semplice rispetto all'uso della console. Gli strumenti da riga di comando possono essere utili se si desidera creare script per eseguire AWS attività.

Quando immetti i comandi in una shell dei comandi, c'è il rischio che la cronologia dei comandi sia accessibile o che le utilità abbiano accesso ai parametri dei comandi. Per informazioni, consulta the section called "Riduci i rischi derivanti dall'utilizzo di per archiviare i tuoi AWS CLI segreti Gestione dei segreti AWS".

Gli strumenti della riga di comando utilizzano automaticamente l'endpoint predefinito per il servizio in una AWS regione. Puoi specificare un endpoint diverso per le tue richieste API. Per informazioni, consulta the section called "Endpoint di Secrets Manager".

AWS fornisce due set di strumenti da riga di comando:

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

AWS SDKs

AWS SDKs Sono costituiti da librerie e codice di esempio per vari linguaggi e piattaforme di programmazione. SDKs Includono attività come la firma crittografica delle richieste, la gestione degli errori e il ritentativo automatico delle richieste. Per scaricare e installare uno qualsiasi di questi SDKs, consulta Tools for Amazon Web Services.

Utilizzano AWS SDKs automaticamente l'endpoint predefinito per il servizio in una AWS regione. Puoi specificare un endpoint diverso per le tue richieste API. Per informazioni, consulta the section called "Endpoint di Secrets Manager".

Per la documentazione SDK, consulta:

- C++
- Go
- Java
- JavaScript
- Kotlin
- .NET
- PHP
- Python (Boto3)
- Ruby
- Rust
- SAP ABAP
- Rapido

API query HTTPS

L'API HTTPS Query offre l'accesso programmatico a Secrets Manager e AWS. L'API Query HTTPS consente di inviare richieste HTTPS direttamente al servizio.

Sebbene sia possibile effettuare chiamate dirette all'API di query HTTPS di Secrets Manager, si consiglia di utilizzare SDKs invece una di queste. L'SDK esegue molte attività utili che altrimenti dovresti effettuare manualmente. Ad esempio, firmano SDKs automaticamente le tue richieste e convertono le risposte in una struttura sintatticamente appropriata alla tua lingua.

Per effettuare chiamate HTTPS a Secrets Manager, ti connetti a ???.

Gestione dei segreti AWS endpoint

Per connettersi a livello di codice a Secrets Manager, si utilizza un endpoint e l'URL del punto di ingresso per il servizio. Gli endpoint Secrets Manager sono endpoint dual-stack, il che significa che supportano entrambi e. IPv4 IPv6

Secrets Manager offre endpoint che supportano gli standard FIPS (Federal Information Processing Standard, standard federali per l'elaborazione delle informazioni) 140-2 in alcune Regioni.

Secrets Manager supporta TLS 1.2 e 1.3. Secrets Manager supporta PQTLS in tutte le Regioni ad eccezione di quelle cinesi.

> **Note**
>
> L' AWS SDK di Python e il AWS CLI tentativo di chiamata IPv6 e quindi IPv4 in sequenza, quindi se non lo hai IPv6 abilitato, potrebbe volerci del tempo prima che la chiamata scada e riprovi. IPv4 Per risolvere questo problema, puoi disabilitarlo IPv6 completamente o migrare a. IPv6

Di seguito sono riportati gli endpoint del servizio per il Secrets Manager. Si noti che la denominazione è diversa dalla tipica convenzione di denominazione dual-stack. Per informazioni sull'utilizzo dell'indirizzamento dual-stack in Secrets Manager, vedere. IPv4 e IPv6 accesso

Nome della regione	Regione	Endpoint	Protocollo	
US East (Ohio)	us-east-2	secretsmanager.us-east-2.amazonaws.com secretsmanager-fips.us-east-2.amazonaws.com	HTTPS HTTPS	
US East (N. Virginia)	us-east-1	secretsmanager.us-east-1.amazonaws.com secretsmanager-fips.us-east-1.amazonaws.com	HTTPS HTTPS	

Nome della regione	Regione	Endpoint	Protocollo	
Stati Uniti occidentali (California settentrionale)	us-west-1	secretsmanager.us-west-1.amazonaws.com secretsmanager-fips.us-west-1.amazonaws.com	HTTPS HTTPS	
Stati Uniti occidentali (Oregon)	us-west-2	secretsmanager.us-west-2.amazonaws.com secretsmanager-fips.us-west-2.amazonaws.com	HTTPS HTTPS	
Africa (Città del Capo)	af-south-1	secretsmanager.af-south-1.amazonaws.com	HTTPS	
Asia Pacifico (Hong Kong)	ap-east-1	secretsmanager.ap-east-1.amazonaws.com	HTTPS	
Asia Pacific (Hyderabad)	ap-south-2	secretsmanager.ap-south-2.amazonaws.com	HTTPS	
Asia Pacifico (Giacarta)	ap-southeast-3	secretsmanager.ap-southeast-3.amazonaws.com	HTTPS	

Nome della regione	Regione	Endpoint	Protocollo	
Asia Pacifico (Malesia)	ap-southeast-5	secretsmanager.ap-southeast-5.amazonaws.com	HTTPS	
Asia Pacifico (Melbourne)	ap-southeast-4	secretsmanager.ap-southeast-4.amazonaws.com	HTTPS	
Asia Pacifico (Mumbai)	ap-south-1	secretsmanager.ap-south-1.amazonaws.com	HTTPS	
Asia Pacifico (Nuova Zelanda)	ap-southeast-6	secretsmanager.ap-southeast-6.amazonaws.com	HTTPS	
Asia Pacifico (Osaka-Locale)	ap-northeast-3	secretsmanager.ap-northeast-3.amazonaws.com	HTTPS	
Asia Pacifico (Seoul)	ap-northeast-2	secretsmanager.ap-northeast-2.amazonaws.com	HTTPS	
Asia Pacifico (Singapore)	ap-southeast-1	secretsmanager.ap-southeast-1.amazonaws.com	HTTPS	

Nome della regione	Regione	Endpoint	Protocollo	
Asia Pacifico (Sydney)	ap-southeast-2	secretsmanager.ap-southeast-2.amazonaws.com	HTTPS	
Asia Pacifico (Taipei)	ap-east-2	secretsmanager.ap-east-2.amazonaws.com	HTTPS	
Asia Pacifico (Thailandia)	ap-southeast-7	secretsmanager.ap-southeast-7.amazonaws.com	HTTPS	
Asia Pacifico (Tokyo)	ap-northeast-1	secretsmanager.ap-northeast-1.amazonaws.com	HTTPS	
Canada (Centrale)	ca-central-1	secretsmanager.ca-central-1.amazonaws.com secretsmanager-fips.ca-central-1.amazonaws.com	HTTPS HTTPS	
Canada occidentale (Calgary)	ca-west-1	secretsmanager.ca-west-1.amazonaws.com secretsmanager-fips.ca-west-1.amazonaws.com	HTTPS HTTPS	
Europa (Francoforte)	eu-central-1	secretsmanager.eu-central-1.amazonaws.com	HTTPS	
Europa (Irlanda)	eu-west-1	secretsmanager.eu-west-1.amazonaws.com	HTTPS	

Nome della regione	Regione	Endpoint	Protocollo	
Europa (Londra)	eu-west-2	secretsmanager.eu-west-2.amazonaws.com	HTTPS	
Europa (Milano)	eu-south-1	secretsmanager.eu-south-1.amazonaws.com	HTTPS	
Europa (Parigi)	eu-west-3	secretsmanager.eu-west-3.amazonaws.com	HTTPS	
Europa (Spagna)	eu-south-2	secretsmanager.eu-south-2.amazonaws.com	HTTPS	
Europa (Stoccolma)	eu-north-1	secretsmanager.eu-north-1.amazonaws.com	HTTPS	
Europa (Zurigo)	eu-central-2	secretsmanager.eu-central-2.amazonaws.com	HTTPS	
Israele (Tel Aviv)	il-central-1	secretsmanager.il-central-1.amazonaws.com	HTTPS	
Messico (Centrale)	mx-central-1	secretsmanager.mx-central-1.amazonaws.com	HTTPS	
Medio Oriente (Bahrein)	me-south-1	secretsmanager.me-south-1.amazonaws.com	HTTPS	
Medio Oriente (Emirati Arabi Uniti)	me-central-1	secretsmanager.me-central-1.amazonaws.com	HTTPS	

Nome della regione	Regione	Endpoint	Protocollo	
Sud America (São Paulo)	sa-east-1	secretsmanager.sa-east-1.amazonaws.com	HTTPS	
AWS GovCloud (Stati Uniti orientali)	us-gov-east-1	secretsmanager.us-gov-east-1.amazonaws.com secretsmanager-fips.us-gov-east-1.amazonaws.com	HTTPS HTTPS	
AWS GovCloud (Stati Uniti occidentali)	us-gov-west-1	secretsmanager.us-gov-west-1.amazonaws.com secretsmanager-fips.us-gov-west-1.amazonaws.com	HTTPS HTTPS	

Gestione dei segreti AWS migliori pratiche

Secrets Manager offre una serie di funzionalità di sicurezza da prendere in considerazione durante lo sviluppo e l'implementazione delle proprie politiche di sicurezza. Le seguenti best practice sono linee guida generali e non rappresentano una soluzione di sicurezza completa. Poiché queste best practice potrebbero non essere appropriate o sufficienti per l'ambiente, sono da considerare come considerazioni utili anziché prescrizioni.

Considerate le seguenti best practice per l'archiviazione e la gestione dei segreti:

- Archivia le credenziali e altre informazioni sensibili in Gestione dei segreti AWS
- Trova segreti non protetti nel tuo codice
- Scegli una chiave di crittografia per il tuo segreto
- Usa la memorizzazione nella cache per recuperare i segreti
- Rotazione dei segreti
- Riduci i rischi legati all'uso della CLI
- Limita l'accesso ai segreti
- Replica i segreti
- Monitorare i segreti
- Gestisci la tua infrastruttura su reti private

Archivia le credenziali e altre informazioni sensibili in Gestione dei segreti AWS

Secrets Manager può contribuire a migliorare il livello di sicurezza e la conformità e a ridurre il rischio di accesso non autorizzato alle informazioni sensibili. Secrets Manager crittografa i segreti inattivi utilizzando chiavi di crittografia di cui l'utente è proprietario e in cui sono archiviate AWS Key Management Service (AWS KMS). Quando recuperate un segreto, Secrets Manager lo decripta e lo trasmette in modo sicuro tramite TLS al vostro ambiente locale. Per ulteriori informazioni, consulta Crea segreti.

Trova segreti non protetti nel tuo codice

CodeGuru Reviewer si integra con Secrets Manager per utilizzare un rilevatore di segreti che trova segreti non protetti nel codice. Il rilevatore di segreti cerca password codificate, stringhe di connessione al database, nomi utente e altro ancora. Per ulteriori informazioni, consulta the section called " CodeGuru Revisore Amazon".

Amazon Q può scansionare la tua base di codice alla ricerca di vulnerabilità di sicurezza e problemi di qualità del codice per migliorare la postura delle tue applicazioni durante l'intero ciclo di sviluppo. Per ulteriori informazioni, consulta Scansione del codice con Amazon Q nella Amazon Q Developer User Guide.

Scegli una chiave di crittografia per il tuo segreto

Nella maggior parte dei casi, consigliamo di utilizzare la chiave `aws/secretsmanager` AWS gestita per crittografare i segreti. Il suo utilizzo non comporta alcun costo.

Per poter accedere a un segreto da un altro account o applicare una politica di chiave alla chiave di crittografia, utilizza una chiave gestita dal cliente per crittografare il segreto.

- Nella politica della chiave, assegna il valore `secretsmanager.<region>.amazonaws.com` alla chiave di `kms:ViaService`condizione. Ciò limita l'uso della chiave solo alle richieste provenienti da Secrets Manager.
- Per limitare ulteriormente l'uso della chiave solo alle richieste di Secrets Manager con il contesto corretto, utilizza chiavi o valori nel contesto di crittografia Secrets Manager come condizione per l'utilizzo della chiave KMS creando:
  - Un operatore di condizione di tipo stringa in una policy IAM o in una policy chiave
  - Un vincolo di concessione in una concessione

Per ulteriori informazioni, consulta the section called "Crittografia e decrittografia del segreto".

Usa la memorizzazione nella cache per recuperare i segreti

Per utilizzare i segreti nel modo più efficiente, si consiglia di utilizzare uno dei seguenti componenti di memorizzazione nella cache di Secrets Manager supportati per memorizzare nella cache i segreti e aggiornarli solo quando necessario:

- Java con memorizzazione nella cache lato client
- Python con memorizzazione nella cache lato client
- .NET con memorizzazione nella cache lato client
- Scegli la memorizzazione nella cache lato client
- Rust con memorizzazione nella cache lato client
- AWS Parametri e segreti: estensione Lambda
- the section called "Amazon EKS"
- Utilizzalo the section called "Agente Secrets Manager" per standardizzare il consumo dei segreti di Secrets Manager in ambienti come AWS Lambda Amazon Elastic Container Service, Amazon Elastic Kubernetes Service e Amazon Elastic Compute Cloud.

Rotazione dei segreti

Se non cambi i tuoi segreti per un lungo periodo di tempo, i segreti possono essere compromessi. Con Secrets Manager, puoi impostare la rotazione automatica ogni quattro ore. Secrets Manager offre due strategie di rotazione: Utente singolo eUtenti alternati. Per ulteriori informazioni, consulta Rotazione dei segreti .

Riduci i rischi legati all'uso della CLI

Quando si utilizza il AWS CLI per richiamare AWS le operazioni, si immettono tali comandi in una shell di comando. La maggior parte delle shell di comando offre funzionalità che potrebbero compromettere i segreti dell'utente, come la registrazione e la possibilità di visualizzare l'ultimo comando immesso. Prima di utilizzare il AWS CLI per inserire informazioni riservate, assicuratevi di farlo. the section called "Riduci i rischi derivanti dall'utilizzo di per archiviare i tuoi AWS CLI segreti Gestione dei segreti AWS"

Limita l'accesso ai segreti

Nelle dichiarazioni politiche di IAM che controllano l'accesso ai tuoi segreti, utilizza il principio dell'accesso con privilegi minimi. Puoi utilizzare i ruoli e le politiche IAM, le politiche delle risorse e il controllo degli accessi basato sugli attributi (ABAC). Per ulteriori informazioni, consulta the section called "Autenticazione e controllo degli accessi".

Argomenti

- Blocca l'ampio accesso ai segreti
- Prestare attenzione alle condizioni degli indirizzi IP nelle politiche
- Limita le richieste con le condizioni degli endpoint VPC

Blocca l'ampio accesso ai segreti

Nelle policy di identità che consentono l'operazione `PutResourcePolicy`, si consiglia di utilizzare `BlockPublicPolicy: true`. Con questa condizione gli utenti possono allegare una policy delle risorse a un segreto se tale policy non consente un accesso ampio.

Secrets Manager utilizza il ragionamento automatizzato di Zelkova per analizzare le policy delle risorse per un accesso ampio. Per ulteriori informazioni su Zelkova, vedi Come AWS utilizza il ragionamento automatico per aiutarti a raggiungere la sicurezza su larga scala sul AWS Security Blog.

L'esempio seguente mostra come utilizzare `BlockPublicPolicy`.

JSON

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

Prestare attenzione alle condizioni degli indirizzi IP nelle politiche

Fai attenzione quando specifichi gli operatori di condizione con indirizzo IP o la chiave di condizione `aws:SourceIp` nella stessa istruzione di policy che consente o rifiuta l'accesso a Secrets Manager.

Ad esempio, se alleghi a un codice segreto una policy che limita AWS le azioni alle richieste provenienti dall'intervallo di indirizzi IP della rete aziendale, le tue richieste come utente IAM che richiama la richiesta dalla rete aziendale funzioneranno come previsto. Tuttavia, se abiliti altri servizi ad accedere al segreto per tuo conto, ad esempio quando abiliti la rotazione con una funzione Lambda, quella funzione richiama le operazioni di Secrets Manager da uno spazio di indirizzi AWS interno. Le richieste influenzate dalla policy con il filtro degli indirizzi IP non vanno a buon fine.

Inoltre, la chiave di condizione `aws:sourceIP` diventa meno efficace quando la richiesta proviene da un endpoint Amazon VPC. Per limitare le richieste a un determinato endpoint VPC, utilizza the section called "Limita le richieste con le condizioni degli endpoint VPC".

Limita le richieste con le condizioni degli endpoint VPC

Per consentire o negare l'accesso alle richieste provenienti da un determinato VPC o endpoint VPC, utilizza `aws:SourceVpc` per limitare l'accesso alle richieste dal VPC specificato o `aws:SourceVpce` per limitare l'accesso alle richieste dall'endpoint VPC specificato. Per informazioni, consulta the section called "Esempio: autorizzazioni e VPCs".

- `aws:SourceVpc` limita l'accesso alle richieste dal VPC specificato.
- `aws:SourceVpce` limita l'accesso alle richieste dall'endpoint VPC specificato.

Se si utilizzano queste chiavi di condizione in una istruzione di policy di risorsa che consente o rifiuta l'accesso ai segreti Secrets Manager puoi inavvertitamente negare l'accesso ai servizi che utilizzano Secrets Manager per accedere ai segreti a tuo nome. Solo alcuni AWS servizi possono essere eseguiti con un endpoint all'interno del tuo VPC. Se limiti le richieste di un segreto a un VPC o endpoint VPC, le chiamate a Secrets Manager da un servizio non configurato per il servizio possono non andare a buon fine.

Per informazioni, consulta the section called "Endpoint VPC (AWS PrivateLink)".

Replica i segreti

Secrets Manager può replicare automaticamente i tuoi segreti in più AWS regioni per soddisfare i tuoi requisiti di resilienza o disaster recovery. Per ulteriori informazioni, consulta Replica in più regioni.

Monitorare i segreti

Secrets Manager consente di controllare e monitorare i segreti attraverso l'integrazione con i servizi AWS di registrazione, monitoraggio e notifica. Per ulteriori informazioni, consulta:

- the section called "Accedi con AWS CloudTrail "
- the section called "Monitora con CloudWatch"
- the section called "Monitora i segreti ai fini della conformità"
- the section called "Monitora i costi di Secrets Manager"
- the section called "Rileva le minacce con GuardDuty"

Gestisci la tua infrastruttura su reti private

Consigliamo di eseguire la maggior parte delle infrastrutture su reti private non accessibili da Internet pubblico. È possibile stabilire una connessione privata tra il VPC e Secrets Manager creando un endpoint VPC dell'interfaccia. Per ulteriori informazioni, consulta the section called "Endpoint VPC (AWS PrivateLink)".

Gestione dei segreti AWS tutorial

Argomenti

Trova segreti non protetti nel tuo codice con Amazon Reviewer CodeGuru

Amazon CodeGuru Reviewer è un servizio che utilizza l'analisi dei programmi e l'apprendimento automatico per rilevare potenziali difetti difficili da trovare per gli sviluppatori e offre suggerimenti per migliorare il codice Java e Python. CodeGuru Reviewer si integra con Secrets Manager per trovare segreti non protetti nel codice. Per i tipi di segreti che riesce a trovare, consulta Tipi di segreti rilevati da CodeGuru Reviewer nella Amazon CodeGuru Reviewer User Guide.

Dopo aver trovato i segreti codificati, completa le seguenti operazioni per sostituirli:

- the section called "Sostituzione delle credenziali del database codificato "
- the section called "Sostituzione dei segreti codificati "

Sposta i segreti codificati in Gestione dei segreti AWS

Se nel codice sono presenti segreti in testo semplice, si consiglia di ruotarli e poi archiviarli in Secrets Manager. Lo spostamento delle credenziali in Secrets Manager risolve il problema di visibilità del segreto a chiunque veda il codice, perché andando avanti il codice recupera il segreto direttamente da Secrets Manager. La rotazione del segreto revoca il segreto codificato corrente in modo che non sia più valido.

Per i segreti delle credenziali del database, consulta Sposta le credenziali del database codificate in Gestione dei segreti AWS.

Prima di iniziare, è necessario determinare chi ha bisogno di accedere al segreto. Consigliamo di utilizzare due ruoli IAM per gestire l'autorizzazione al tuo segreto:

- Un ruolo che gestisce i segreti nella tua organizzazione. Per ulteriori informazioni, consulta the section called "Autorizzazioni di amministrazione di Secrets Manager". Questo ruolo sarà utilizzato per creare e ruotare il segreto.
- Un ruolo che può utilizzare il segreto in fase di esecuzione, ad esempio in questo tutorial che usi. *RoleToRetrieveSecretAtRuntime* Il tuo codice assume questo ruolo per recuperare il segreto. In questo tutorial, si concede al ruolo solo l'autorizzazione per recuperare il valore di un segreto e si concede l'autorizzazione utilizzando la policy delle risorse del segreto. Per le alternative, consulta the section called "Fasi successive".

Fasi:

- Fase 1: creazione del segreto
- Fase 2: aggiornamento del codice
- Fase 3: aggiornamento del segreto
- Fasi successive

Fase 1: creazione del segreto

Il primo passo consiste nel copiare il segreto codificato esistente in Secrets Manager. Se il segreto è correlato a una AWS risorsa, memorizzalo nella stessa regione della risorsa. Altrimenti, archivialo nella regione con la latenza più bassa per il tuo caso d'uso.

Creazione di un segreto (console)

1. Apri la console Secrets Manager all'indirizzo https://console.aws.amazon.com/secretsmanager/.
2. Scegli Archivia un nuovo segreto.
3. Nella pagina Choose secret type (Scegli il tipo di segreto), effettua le seguenti operazioni:

   a. Per Secret type (Tipo di segreto), scegli Other type of secret (Altro tipo di segreto).

   b. Inserisci il tuo segreto come coppie chiave/valore o in testo semplice. Alcuni esempi:

      API key

         Entra in key/value coppia:

ClientID : *my_client_id*

ClientSecret : *wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY*

OAuth token

Inserisci come testo semplice:

AKIAI44QH8DHBEXAMPLE

Digital certificate

Inserisci come testo semplice:

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

Private key

Inserisci come testo semplice:

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

c. In Encryption key (Chiave di crittografia), scegli aws/secretsmanager per utilizzare la Chiave gestita da AWS per Secrets Manager. L'utilizzo di questa chiave non prevede costi aggiuntivi. Puoi inoltre utilizzare la tua chiave gestita dal cliente, ad esempio per accedere al segreto da un altro Account AWS. Per informazioni sui costi di utilizzo di una chiave gestita dal cliente, consulta la sezione Prezzi.

d. Scegli Next (Successivo).

4. Nella pagina Choose secret type (Scegli il tipo di segreto), effettua le seguenti operazioni:

a. Inserisci un Secret name (Nome del segreto) e una Description (Descrizione) descrittivi.

b. In Resource permissions (Autorizzazioni della risorsa), scegli Edit permissions (Modifica autorizzazioni). Incolla la seguente politica, che consente di ***RoleToRetrieveSecretAtRuntime*** recuperare il segreto, quindi scegli Salva.

JSON

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

c. Nella parte inferiore della pagina scegli Next (Avanti).

5. Nella pagina Configure rotation (Configura la rotazione), mantieni la rotazione disattivata. Scegli Next (Successivo).
6. Nella pagina Review (Revisione), rivedi i dettagli dei segreti e quindi scegli Store (Archivia).

Fase 2: aggiornamento del codice

Il codice deve assumere il ruolo IAM *RoleToRetrieveSecretAtRuntime* per poter recuperare il segreto. Per ulteriori informazioni, consulta Passare a un ruolo IAM (AWS API).

Successivamente, aggiorna il codice per recuperare il segreto da Secrets Manager utilizzando il codice di esempio fornito da Secrets Manager.

Ricerca del codice di esempio

1. Apri la console Secrets Manager all'indirizzo https://console.aws.amazon.com/secretsmanager/.
2. Nella pagina dell'elenco Secrets (Segreti) scegli il segreto.
3. Scorrere verso il basso fino a Sample code (Codice di esempio). Scegli il linguaggio di programmazione, quindi copia il frammento di codice.

Nell'applicazione, rimuovi il segreto codificato e incolla il frammento di codice. A seconda del linguaggio del codice, potrebbe essere necessario aggiungere una chiamata alla funzione o al metodo nel frammento.

Verifica che la tua applicazione funzioni come previsto con il segreto al posto del segreto codificato.

Fase 3: aggiornamento del segreto

L'ultima fase è la revoca e l'aggiornamento del segreto codificato. Fai riferimento all'origine del segreto per trovare le istruzioni per revocare e aggiornare il segreto. Ad esempio, potrebbe essere necessario disattivare il segreto corrente e generarne uno nuovo.

Aggiornamento del segreto con il nuovo valore

1. Apri la console Secrets Manager all'indirizzo https://console.aws.amazon.com/secretsmanager/.
2. Scegli Secrets (Segreti), quindi scegli il segreto.
3. Nella pagina Secret details (Dettagli del segreto), scorri verso il basso e scegli Retrieve secret value (Recupera il valore del segreto), quindi Edit (Modifica).
4. Aggiorna il segreto, quindi scegli Save (Salva).

Quindi, verifica che la tua applicazione funzioni come previsto con il nuovo segreto.

Fasi successive

Dopo aver rimosso un segreto codificato dal codice, ecco alcuni concetti da considerare:

- Per trovare segreti codificati nelle tue applicazioni Java e Python, ti consigliamo Amazon Reviewer. CodeGuru
- È possibile migliorare le prestazioni e ridurre i costi tramite la memorizzazione nella cache dei segreti. Per ulteriori informazioni, consulta Ottieni segreti.
- Per i segreti a cui accedi da più regioni, considera la possibilità di replicare il tuo segreto per migliorare la latenza. Per ulteriori informazioni, consulta Replica in più regioni.
- In questo tutorial, hai concesso `RoleToRetrieveSecretAtRuntime` solo l'autorizzazione per recuperare il valore segreto. Per concedere al ruolo più autorizzazioni, ad esempio per ottenere metadati sul segreto o per visualizzare un elenco di segreti, consulta the section called "Policy basate sulle risorse".

- In questo tutorial, hai concesso l'autorizzazione *RoleToRetrieveSecretAtRuntime* utilizzando la politica delle risorse del segreto. Per altri modi per concedere l'autorizzazione, consulta the section called "Policy basate sull'identità".

Sposta le credenziali del database codificate in Gestione dei segreti AWS

Se nel codice sono presenti credenziali di database in testo semplice, si consiglia di spostare le credenziali in Secrets Manager e quindi ruotarle immediatamente. Lo spostamento delle credenziali in Secrets Manager risolve il problema di visibilità delle credenziali a chiunque veda il codice, perché andando avanti il codice recupera le credenziali direttamente da Secrets Manager. La rotazione del segreto aggiorna la password e quindi revoca la password codificata corrente in modo che non sia più valida.

Per i database Amazon RDS, Amazon Redshift e Amazon DocumentDB, utilizza la procedura descritta in questa pagina per spostare le credenziali codificate in Secrets Manager. Per altri tipi di credenziali e altri segreti, consulta the section called "Sostituzione dei segreti codificati ".

Prima di iniziare, è necessario determinare chi ha bisogno di accedere al segreto. Consigliamo di utilizzare due ruoli IAM per gestire l'autorizzazione al tuo segreto:

- Un ruolo che gestisce i segreti nella tua organizzazione. Per ulteriori informazioni, consulta the section called "Autorizzazioni di amministrazione di Secrets Manager". Questo ruolo sarà utilizzato per creare e ruotare il segreto.
- Un ruolo che può utilizzare le credenziali in fase di esecuzione, *RoleToRetrieveSecretAtRuntime* in questo tutorial. Il tuo codice assume questo ruolo per recuperare il segreto.

Fasi:

- Fase 1: creazione del segreto
- Fase 2: aggiornamento del codice
- Fase 3: rotazione del segreto
- Fasi successive

Fase 1: creazione del segreto

Il primo passo consiste nel copiare le credenziali codificate esistenti in un segreto in Secrets Manager. Per la latenza più bassa, archivia il segreto nella stessa regione del database.

Per creare un segreto

1. Apri la console Secrets Manager all'indirizzo https://console.aws.amazon.com/secretsmanager/.
2. Scegli Archivia un nuovo segreto.
3. Nella pagina Choose secret type (Scegli il tipo di segreto), effettua le seguenti operazioni:
   a. Per Secret type (Tipo di segreto), scegli il tipo di credenziali del database da archiviare:
      - Database Amazon RDS
      - Database di Amazon DocumentDB
      - Data warehouse Amazon Redshift.
      - Per altri tipi di segreti, consulta Sostituzione dei segreti codificati.
   b. Per Credenziali, inserisci le credenziali codificate per il database.
   c. In Encryption key (Chiave di crittografia), scegli aws/secretsmanager per utilizzare la Chiave gestita da AWS per Secrets Manager. L'utilizzo di questa chiave non prevede costi aggiuntivi. Puoi inoltre utilizzare la tua chiave gestita dal cliente, ad esempio per accedere al segreto da un altro Account AWS. Per informazioni sui costi di utilizzo di una chiave gestita dal cliente, consulta la sezione Prezzi.
   d. Per Database, scegli il database.
   e. Scegli Next (Successivo).
4. Nella pagina Configure secret (Configura il segreto), effettua le seguenti operazioni:
   a. Inserisci un Secret name (Nome del segreto) e una Description (Descrizione) descrittivi.
   b. In Resource permissions (Autorizzazioni della risorsa), scegli Edit permissions (Modifica autorizzazioni). Incolla la seguente policy, che ***RoleToRetrieveSecretAtRuntime*** consente di recuperare il segreto, quindi scegli Salva.

      JSON

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

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

c. Nella parte inferiore della pagina scegli Next (Avanti).

5. Nella pagina Configure rotation (Configura la rotazione), mantieni la rotazione disattivata per il momento. La attiverai più tardi. Scegli Next (Successivo).
6. Nella pagina Review (Revisione), rivedi i dettagli dei segreti e quindi scegli Store (Archivia).

Fase 2: aggiornamento del codice

Il codice deve assumere il ruolo IAM *RoleToRetrieveSecretAtRuntime* per poter recuperare il segreto. Per ulteriori informazioni, consulta Passare a un ruolo IAM (AWS API).

Successivamente, aggiorna il codice per recuperare il segreto da Secrets Manager utilizzando il codice di esempio fornito da Secrets Manager.

Ricerca del codice di esempio

1. Apri la console Secrets Manager all'indirizzo https://console.aws.amazon.com/secretsmanager/.
2. Nella pagina dell'elenco Secrets (Segreti) scegli il segreto.
3. Scorrere verso il basso fino a Sample code (Codice di esempio). Scegli il linguaggio, quindi copia il frammento di codice.

Nell'applicazione, rimuovi le credenziali codificate e incolla il frammento di codice. A seconda del linguaggio del codice, potrebbe essere necessario aggiungere una chiamata alla funzione o al metodo nel frammento.

Verifica che la tua applicazione funzioni come previsto con il segreto al posto delle credenziali codificate.

Fase 3: rotazione del segreto

L'ultimo passo è revocare le credenziali codificate ruotando il segreto. La rotazione è il processo di aggiornamento periodico di un segreto. Quando si ruota un segreto, vengono aggiornate le credenziali sia nel segreto che nel database. Secrets Manager può ruotare automaticamente un segreto su un programma configurato.

Parte del processo di configurazione della rotazione è garantire che la funzione di rotazione Lambda possa accedere sia a Secrets Manager che al database. Quando si attiva la rotazione automatica, Secrets Manager crea la funzione di rotazione Lambda nello stesso VPC del database in modo che abbia accesso di rete al database. La funzione di rotazione Lambda deve anche essere in grado di effettuare chiamate a Secrets Manager per aggiornare il segreto. Ti consigliamo di creare un endpoint Secrets Manager nel VPC in modo che le chiamate da Lambda a Secrets Manager non lascino l'infrastruttura. AWS Per istruzioni, consulta the section called “Endpoint VPC (AWS PrivateLink)”.

Attivazione della rotazione

1. Apri la console Secrets Manager all'indirizzo https://console.aws.amazon.com/secretsmanager/.
2. Nella pagina dell'elenco Secrets (Segreti) scegli il segreto.
3. Nella pagina Secret details (Dettagli del segreto), nella sezione Rotation configuration (Configurazione rotazione) scegli Edit rotation (Modifica rotazione).
4. Nella finestra di dialogo Edit rotation configuration (modifica configurazione rotazione), procedi come indicato di seguito:
    a. Attiva Automatic rotation (Rotazione automatica).
    b. In Rotation schedule (Programma di rotazione), inserisci il programma nel fuso orario UTC.
    c. Scegli Rotate immediately when the secret is stored (Ruota immediatamente quando viene memorizzato il segreto) per ruotare il segreto al salvataggio delle modifiche.
    d. In Rotation function (Funzione di rotazione), scegli Create a new Lambda function (Crea una nuova funzione Lambda) ed immetti un nome per la nuova funzione. Secrets Manager aggiunge "SecretsManager" all'inizio del nome della funzione.
    e. Per Strategia di rotazione scegli Utente singolo.
    f. Scegli Save (Salva).

Verifica che il segreto sia stato ruotato

1. Apri la console Secrets Manager all'indirizzo https://console.aws.amazon.com/secretsmanager/.
2. Scegli Secrets (Segreti), quindi scegli il segreto.
3. Nella pagina Secret details (Dettagli del segreto), scorri e scegli Retrieve secret value (Recupera il valore del segreto).

   Se il valore del segreto è cambiato, allora la rotazione è riuscita. Se il valore segreto non è cambiato, devi farlo Risoluzione dei problemi della rotazione guardando CloudWatch i log per la funzione di rotazione.

Verifica che la tua applicazione funzioni come previsto con il segreto ruotato.

Fasi successive

Dopo aver rimosso un segreto codificato dal codice, ecco alcuni concetti da considerare:

- È possibile migliorare le prestazioni e ridurre i costi tramite la memorizzazione nella cache dei segreti. Per ulteriori informazioni, consulta Ottieni segreti.
- È possibile scegliere un diverso programma di rotazione. Per ulteriori informazioni, consulta the section called “Pianificazioni di rotazione”.
- Per trovare segreti codificati nelle tue applicazioni Java e Python, ti consigliamo Amazon Reviewer. CodeGuru

Imposta la rotazione alternata degli utenti per Gestione dei segreti AWS

In questo tutorial, imparerai come impostare la rotazione a utenti alternati per un segreto che contiene le credenziali del database. La rotazione a utenti alternati è una strategia di rotazione in cui Secrets Manager clona l'utente e quindi alterna quali credenziali dell'utente vengono aggiornate. Questa strategia è una buona scelta se hai bisogno di disponibilità elevata per il segreto, perché uno degli utenti alternati ha le credenziali correnti per il database mentre l'altro è in fase di aggiornamento. Per ulteriori informazioni, consulta the section called “Utenti alternati”.

Per impostare la rotazione a utenti alternati, sono necessari due segreti:

- Un segreto con le credenziali da ruotare.

- Un secondo segreto con credenziali di amministratore.

  Questo utente dispone delle autorizzazioni per clonare il primo utente e modificare la password del primo utente. In questo tutorial, chiedi ad Amazon RDS di creare questo segreto per un utente amministratore. Amazon RDS gestisce anche la rotazione delle password dell'amministratore. Per ulteriori informazioni, consulta the section called "Rotazione gestita".

La prima parte di questo tutorial tratta la configurazione di un ambiente realistico. Per mostrare come funziona la rotazione, questo tutorial utilizza un esempio di database Amazon RDS MySQL. Per motivi di sicurezza, il database si trova in un VPC che limita l'accesso a Internet. Per connettersi al database dal computer locale tramite Internet, è necessario utilizzare un host bastione, un server nel VPC in grado di connettersi al database, ma che consente anche connessioni SSH da Internet. L'host bastione in questo tutorial è un'istanza Amazon EC2 e i gruppi di sicurezza per l'istanza impediscono altri tipi di connessioni.

Al termine del tutorial, si consiglia di ripulire le risorse del tutorial. Non utilizzarle in un ambiente di produzione.

La rotazione di Secrets Manager utilizza una AWS Lambda funzione per aggiornare il segreto e il database. Per ulteriori informazioni sui costi di utilizzo della funzione Lambda, consulta la sezione Prezzi.

Tutorial:

Permissions

Per i prerequisiti del tutorial, hai bisogno di autorizzazioni di amministrazione per il tuo Account AWS. In un ambiente di produzione, è consigliabile utilizzare ruoli diversi per ciascun passaggio.

Ad esempio, un ruolo con le autorizzazioni dell'amministratore del database creerebbe il database Amazon RDS e un ruolo con autorizzazioni di amministrazione di rete configurerebbe il VPC e i gruppi di sicurezza. Per i passaggi del tutorial, suggeriamo di continuare a utilizzare la stessa identità.

Per informazioni su come configurare le autorizzazioni in un ambiente di produzione, consulta the section called "Autenticazione e controllo degli accessi".

Prerequisiti

Per questo tutorial hai bisogno dei seguenti elementi:

- Prerequisito A: Amazon VPC
- Prerequisito B: istanza Amazon EC2
- Prerequisito C: database Amazon RDS e un segreto in Secrets Manager per le credenziali di amministratore
- Prerequisito D: consenti al computer locale di connettersi all'istanza EC2.

Prerequisito A: Amazon VPC

In questa fase viene creato un VPC in cui è necessario avviare un database Amazon RDS e un'istanza Amazon EC2. In una fase successiva, utilizzerai il tuo computer per connetterti tramite Internet all'host bastione e al database, quindi dovrai consentire al traffico di uscire dal VPC. Per fare ciò, Amazon VPC collega un gateway Internet al gateway Internet e aggiunge un routing alla tabella di routing in modo che il traffico destinato al di fuori del VPC venga inviato al gateway Internet.

All'interno del VPC, crei un endpoint Secrets Manager e un endpoint Amazon RDS. Quando si imposta la rotazione automatica in una fase successiva, Secrets Manager crea la funzione di rotazione Lambda all'interno del VPC in modo che abbia accesso al database. La funzione di rotazione Lambda chiama anche Secrets Manager per aggiornare il segreto e chiama Amazon RDS per ottenere le informazioni di connessione al database. Creando endpoint all'interno del VPC, ti assicuri che le chiamate dalla funzione Lambda a Secrets Manager e Amazon RDS non lascino l'infrastruttura. AWS Invece, vengono instratade all'endpoint all'interno del VPC.

Per creare un VPC

1. Apri la console Amazon VPC all'indirizzo https://console.aws.amazon.com/vpc/.
2. Seleziona Crea VPC.
3. Nella pagina Create VPC (Crea VPC), scegli VPC and more (VPC e altro).

4. In Name tag auto-generation (Generazione automatica del tag del nome), sotto a Auto-generate (genera automaticazione), inserisci **`SecretsManagerTutorial`**.
5. Per le opzioni DNS, scegli entrambi **`Enable DNS hostnames`** e **`Enable DNS resolution`**.
6. Seleziona Crea VPC.

Creare un endpoint di Secrets Manager all'interno del VPC

1. Nella console Amazon VPC, sotto Endpoints, scegli Create endpoint (crea endpoint).
2. In Endpoint settings (Impostazioni endpoint), per Name (Nome) inserisci **`SecretsManagerTutorialEndpoint`**.
3. Sotto a Services (Servizi), inserisci **`secretsmanager`** per filtrare l'elenco e quindi seleziona l'endpoint di Secrets Manager nella Regione AWS. Per esempio, negli Stati Uniti orientali (Virginia settentrionale), scegli `com.amazonaws.us-east-1.secretsmanager`.
4. Per VPC, scegli **`vpc**** (SecretsManagerTutorial)`**.
5. Per Subnets (Sottoreti), seleziona tutte le Availability Zones (Zone di disponibilità) e poi per ognuna di esse scegli un Subnet ID (ID sottorete) da includere.
6. Per Tipo di indirizzo IP, scegli **`IPv4`**.
7. Da Security Groups (Gruppi di sicurezza), scegli il gruppo di sicurezza di default.
8. Per Policy type (Tipo di policy), scegli **`Full access`**.
9. Seleziona Crea endpoint.

Creare un endpoint di Amazon RDS all'interno del VPC

1. Nella console Amazon VPC, sotto Endpoints, scegli Create endpoint (crea endpoint).
2. In Endpoint settings (Impostazioni endpoint), per Name (Nome) inserisci **`RDSTutorialEndpoint`**.
3. Sotto a Services (Servizi), inserisci **`rds`** per filtrare l'elenco e quindi seleziona l'endpoint di Amazon RDS in Regione AWS. Per esempio, negli Stati Uniti orientali (Virginia settentrionale), scegli `com.amazonaws.us-east-1.rds`.
4. Per VPC, scegli **`vpc**** (SecretsManagerTutorial)`**.
5. Per Subnets (Sottoreti), seleziona tutte le Availability Zones (Zone di disponibilità) e poi per ognuna di esse scegli un Subnet ID (ID sottorete) da includere.
6. Per Tipo di indirizzo IP, scegli **`IPv4`**.

7. Da Security Groups (Gruppi di sicurezza), scegli il gruppo di sicurezza di default.
8. Per Policy type (Tipo di policy), scegli **`Full access`**.
9. Seleziona Crea endpoint.

Prerequisito B: istanza Amazon EC2

Il database Amazon RDS che creerai in una fase successiva sarà nel cloud VPC, per accedervi è necessario un host bastione. L'host bastione è presente anche nel VPC, ma in una fase successiva, configurerai un gruppo di sicurezza per consentire al computer locale di connettersi all'host bastione con SSH.

Creare un'istanza EC2 per un host bastione

1. Apri la console Amazon EC2 all'indirizzo https://console.aws.amazon.com/ec2/.
2. Scegliere Instances (Istanze), quindi scegliere Launch Instances (Avvia istanze).
3. Nell'area Name and tags (Nome e tag), in Name (Nome) inserisci **`SecretsManagerTutorialInstance`**.
4. In Application and OS Images (Immagini dell'applicazione e del sistema operativo), mantieni l'impostazione predefinita **`Amazon Linux 2 AMI (HMV) Kernel 5.10`**.
5. In Instance type (Tipo di istanza), mantieni l'impostazione predefinita **`t2.micro`**.
6. In Key pair (coppia di chiavi), scegli Crea coppia di chiavi.

   Nella finestra di dialogo Create Key Pair (Crea coppia di chiavi), per Key pair name (Nome coppia di chiavi), inserisci **`SecretsManagerTutorialKeyPair`**, quindi scegli Create (Crea).

   La chiave privata viene scaricata automaticamente.
7. In Network settings (Impostazioni di rete), scegli Edit (Modifica) ed esegui le operazioni qui descritte:

   a. Per VPC, scegliere **`vpc-**** SecretsManagerTutorial`**.

   b. Per Auto-assign Public IP (Assegna automaticamente IP pubblico), scegliere **`Enable`**.

   c. Per Firewall, scegli Select existing security group (seleziona gruppo di sicurezza esistente).

   d. Per i gruppi di sicurezza comuni, scegli **`default`**.
8. Scegliere Launch Instance (Avvia istanza).

Prerequisito C: database Amazon RDS e un segreto in Secrets Manager per le credenziali di amministratore

In questa fase, crei un database Amazon MySQL e lo configuri in modo che Amazon RDS crei un segreto per contenere le credenziali di amministratore. Quindi Amazon RDS gestisce automaticamente la rotazione del segreto di amministratore per te. Per ulteriori informazioni, consulta Rotazione gestita.

Durante la creazione del database, è necessario specificare l'host bastione creato nel passaggio precedente. Quindi Amazon RDS configura i gruppi di sicurezza in modo che il database e l'istanza possano accedere vicendevolmente. Si aggiunge una regola al gruppo di sicurezza collegato all'istanza per consentire anche al computer locale di connettersi ad essa.

Per creare un database Amazon RDS con un segreto di Secrets Manager contenente le credenziali di amministratore

1. Nella console Amazon RDS scegli Create databases (Crea database).
2. Nella sezione Engine options (Opzioni motore) per Engine type (tipo motore) scegli **`MySQL`**.
3. Nella sezione Templates (Modelli), seleziona **`Free tier`**.
4. Nella sezione Rule settings (Impostazioni regole), procedi nel seguente modo:
    a. Per l'identificatore dell'istanza DB, inserisci **`SecretsManagerTutorial`**.
    b. In Impostazioni delle credenziali, seleziona Gestisci le credenziali principali in. Gestione dei segreti AWS
5. Nella sezione Connectivity (Connettività), per Computer resource (Risorsa computer), scegli Connect to an EC2 computer resource (Connetti a una risorsa computer EC2) e quindi, per l'istanza EC2, scegli **`SecretsManagerTutorialInstance`**.
6. Scegliere Crea database.

Prerequisito D: consenti al computer locale di connettersi all'istanza EC2.

In questo passaggio, configuri l'istanza EC2 creata nel Prerequisito B per consentire al computer locale di connettersi ad essa. A tale scopo, modifichi il gruppo di sicurezza aggiunto da Amazon RDS in Prereq C per includere una regola che consenta all'indirizzo IP del tuo computer di connettersi a SSH. La regola consente al computer locale (identificato dal tuo attuale indirizzo IP) di connettersi all'host bastione utilizzando SSH su Internet.

Prerequisito D: consenti al computer locale di connettersi all'istanza EC2.

1. Apri la console Amazon EC2 all'indirizzo https://console.aws.amazon.com/ec2/.
2. Nell'istanza EC2 SecretsManagerTutorialInstance, nella scheda Sicurezza, in Gruppi di sicurezza, scegli. **`sg-*** (ec2-rds-X)`**
3. Sotto a Inbound rules (Regole in entrata), seleziona Edit inbound rules (Modifica regole in entrata).
4. Scegli Add rule (Aggiungi regola), poi per la regola esegui le seguenti operazioni:
   a. In Tipo, scegli **`SSH`**.
   b. Per Source (origine), scegli **`My IP`**.

Fase 1: creazione di un utente del database Amazon RDS

Innanzitutto, è necessario un utente le cui credenziali saranno memorizzate nel segreto. Per creare l'utente, accedi al database Amazon RDS con le credenziali di amministratore. Per semplicità, nel tutorial, si crea un utente con piena autorizzazione per un database. In un ambiente di produzione questo non succede normalmente e ti consigliamo di seguire il principio del privilegio minimo.

Per connettersi al database, si utilizza uno strumento client MySQL. In questa esercitazione, sarà possibile usare MySQL Workbench, un'applicazione basata su GUI. Scaricare e installare MySQL Workbench dalla pagina di Download MySQL Workbench (Scarica MySQL Workbench).

Per connettersi al database, è necessario creare una configurazione di connessione in MySQL Workbench. Per la configurazione, sono necessarie alcune informazioni sia da Amazon EC2 che Amazon RDS.

Creare una connessione al database in MySQL Workbench

1. In MySQL Workbench, accanto a MySQL Connections (Connessioni MySQL), scegli il pulsante (+).
2. Nella finestra di dialogo Setup New Connection (Configura una nuova connessione), segui questi passaggi:
   a. Per Connection Name (Nome connessione), inserisci **`SecretsManagerTutorial`**.
   b. Per Connection Method (Metodo di connessione), scegli **`Standard TCP/IP over SSH`**.
   c. Nella scheda Parameters (Parametri), procedi come segue:

i. Per Hostname SSH (Nome host SSH), inserisci l'indirizzo IP pubblico dell'istanza Amazon EC2.

Puoi trovare l'indirizzo IP sulla console Amazon EC2 scegliendo l'istanza. SecretsManagerTutorialInstance Copia l'indirizzo IP in Public IPv4 DNS.

ii. Per SSH Username (Nome utente SSH), inserisci **`ec2-user`**.

iii. Per SSH Keyfile, scegli il file di coppia di chiavi SecretsManagerTutorialKeyPair.pem che hai scaricato nel prerequisito precedente.

iv. Per MySQL Hostname (Home host MySQL), inserisci l'indirizzo dell'endpoint Amazon RDS.

Puoi trovare l'indirizzo endpoint sulla console Amazon RDS scegliendo l'istanza di database secretsmanagertutorialdb. Copia l'indirizzo in Endpoint.

v. Per Username (Nome utente), inserisci **`admin`**.

d. Scegli OK.

Recuperare la password dell'amministratore

1. Nella console Amazon RDS scegli il database.
2. Nella scheda Configuration (Configurazione), in Master Credentials ARN (ARN delle credenziali principali), scegli Manage in Secrets Manager (gestisci in secrets manager).

   Si apre la console Secrets Manager.
3. Nella pagina dei dettagli del segreto, scegli Retrieve secret value (Recupera il valore di un segreto).
4. La password viene visualizzata nella sezione Secret value (valore segreto).

Per creare un utente del database

1. In MySQL Workbench, scegli la connessione. SecretsManagerTutorial
2. Inserisci la password dell'amministratore che hai recuperato dal segreto.
3. In MySQL Workbench, nella finestra Query, inserisci i seguenti comandi (inclusa una password sicura) e quindi scegli Execute (Esegui). La funzione di rotazione verifica il segreto aggiornato utilizzando SELECT, quindi **`appuser`** deve avere almeno quel privilegio.

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

Nella finestra Output, viene visualizzato l'esito positivo dei comandi.

Fase 2: creazione di un segreto per le credenziali dell'utente

Successivamente, creerai un segreto per archiviare le credenziali dell'utente appena creato. Questo è il segreto che verrà ruotato. Attiva la rotazione automatica e, per indicare la strategia a utenti alternati, scegli un segreto di un utente con privilegi avanzati separato che dispone dell'autorizzazione per modificare la password del primo utente.

1. Apri la console Secrets Manager all'indirizzo https://console.aws.amazon.com/secretsmanager/.
2. Scegli Archivia un nuovo segreto.
3. Nella pagina Choose secret type (Scegli il tipo di segreto), effettua le seguenti operazioni:
   a. Per Secret type (Tipo segreto), scegli Credentials for Amazon RDS database (Credenziali per il database Amazon RDS).
   b. Per Credentials (Credenziali), inserisci il nome utente **appuser** e la password inseritaper l'utente del database creato utilizzando MySQL Workbench.
   c. Per Database, scegli secretsmanagertutorialdb.
   d. Scegli Next (Successivo).
4. Nella pagina Configure secret (Configura il segreto), per Secret name (Nome del segreto), inserisci **SecretsManagerTutorialAppuser** e scegli Next (Successivo).
5. Nella pagina Configure rotation (Configura la rotazione), effetua le seguenti operazioni:
   a. Attiva Automatic rotation (Rotazione automatica).
   b. Per Rotation schedule (Pianificazione della rotazione), imposta una pianificazione di Days (Giorni): **2** Days with Duration: **2h** (2 giorni con durata: 2 h). Mantieni selezionato Rotate immediately (Ruota immediatamente).
   c. Per Rotation function (Funzione di rotazione), scegli Create a rotation function (Crea una funzione di rotazione) e per il nome della funzione inserisci **tutorial-alternating-users-rotation**.

d. Per Strategia di rotazione, scegli Utenti alternati, quindi in Segreto credenziali amministratore scegli il segreto denominato rds!cluster... che ha una Descrizione che include il nome del database creato in questo tutorial **`secretsmanagertutorial`**, ad esempio `Secret associated with primary RDS DB instance: arn:aws:rds:`*`Region`*`:`*`AccountId`*`:db:secretsmanagertutorial`.

e. Scegli Next (Successivo).

6. Nella pagina Review (Rivedi), scegli Store (Archivia).

   Secrets Manager torna alla pagina dei dettagli segreti. Nella parte superiore della pagina, è possibile visualizzare lo stato della configurazione di rotazione. Secrets Manager utilizza CloudFormation per creare risorse come la funzione di rotazione Lambda e un ruolo di esecuzione che esegue la funzione Lambda. Al CloudFormation termine, il banner diventa Segreto programmato per la rotazione. La prima rotazione è completa.

Fase 3: eseguire il test del segreto ruotato

Ora che il segreto è stato ruotato, è possibile verificare che il segreto contenga ancora credenziali valide. La password nel segreto è cambiata rispetto alle credenziali originali.

Recuperare la nuova password dal segreto

1. Apri la console Secrets Manager all'indirizzo https://console.aws.amazon.com/secretsmanager/.
2. Scegli Secrets (Segreti) e quindi scegli il segreto **`SecretsManagerTutorialAppuser`**.
3. Nella pagina Secret details (Dettagli del segreto), scorri e scegli Retrieve secret value (Recupera il valore del segreto).
4. Nella tabella Key/value (Chiave/valore), copia il Secret value (Valore segreto) per la **`password`**.

Testare le credenziali

1. In MySQL Workbench, fai clic con il pulsante destro del mouse sulla SecretsManagerTutorialconnessione e quindi scegli Modifica connessione.
2. Nella finestra di dialogo Manage Server Connections (Gestisci connessioni al server), per Username (Nome utente), inserisci **`appuser`** e scegli Close (Chiudi).
3. Tornando in MySQL Workbench, scegli la connessione. SecretsManagerTutorial

4. Nella finestra di dialogo Open SSH Connection (Apri connessione SSH), per Password incolla la password recuperata dal segreto, quindi scegli OK.

   Se le credenziali sono valide, MySQL Workbench si apre alla pagina di progettazione del database.

Questa mostra che la rotazione del segreto ha esito positivo. Le credenziali nel segreto sono state aggiornate e consistono in una password valida per connettersi al database.

Fase 4: Eliminazione delle risorse

Per provare un'altra strategia di rotazione, la rotazione a utente singolo, salta la pulizia delle risorse e vai a the section called "Rotazione a utente singolo".

Per evitare potenziali addebiti e per rimuovere l'istanza EC2 che ha accesso a Internet, elimina le seguenti risorse create in questo tutorial e i relativi prerequisiti:

- Istanza database Amazon RDS Per istruzioni, consulta Eliminazione di un'istanza DB nella Guida per l'utente di Amazon RDS.
- Istanza Amazon EC2. Per istruzioni, consulta Terminare un'istanza nella Guida per l'utente di Amazon EC2.
- Segreto di Secrets Manager `SecretsManagerTutorialAppuser`. Per istruzioni, consulta the section called "Eliminare un segreto".
- Endpoint di Secrets Manager. Per istruzioni, consulta Eliminazione di un endpoint VPC nella Guida AWS PrivateLink .
- Endpoint VPC. Per istruzioni, consulta Eliminazione di un VPC nella Guida AWS PrivateLink .

Fasi successive

- Ottieni informazioni su come recuperare i segreti nelle applicazioni.
- Scopri altri programmi di rotazione.

Imposta la rotazione di un singolo utente per Gestione dei segreti AWS

In questo tutorial, imparerai come impostare la rotazione a utente singolo per un segreto che contiene le credenziali del database. La rotazione a utente singolo è una strategia di rotazione in cui Secrets Manager aggiorna le credenziali di un utente sia nel segreto che nel database. Per ulteriori informazioni, consulta the section called "Utente singolo".

Al termine del tutorial, si consiglia di ripulire le risorse del tutorial. Non utilizzarle in un ambiente di produzione.

La rotazione di Secrets Manager utilizza una AWS Lambda funzione per aggiornare il segreto e il database. Per ulteriori informazioni sui costi di utilizzo della funzione Lambda, consulta la sezione Prezzi.

Indice

Permissions

Per i prerequisiti del tutorial, hai bisogno di autorizzazioni di amministrazione per il tuo Account AWS. In un ambiente di produzione, è consigliabile utilizzare ruoli diversi per ciascun passaggio. Ad esempio, un ruolo con le autorizzazioni dell'amministratore del database creerebbe il database Amazon RDS e un ruolo con autorizzazioni di amministrazione di rete configurerebbe il VPC e i gruppi di sicurezza. Per i passaggi del tutorial, suggeriamo di continuare a utilizzare la stessa identità.

Per informazioni su come configurare le autorizzazioni in un ambiente di produzione, consulta the section called "Autenticazione e controllo degli accessi".

Prerequisiti

Il prerequisito per questo tutorial è the section called "Rotazione a utenti alternati". Non eliminare le risorse alla fine del primo tutorial. Dopo questo tutorial, disporrai di un ambiente realistico con un database Amazon RDS e un segreto di Secrets Manager che contiene le credenziali di amministratore per il database. Hai anche un secondo segreto che contiene le credenziali per un utente del database, ma non lo usi in questo tutorial.

È inoltre configurata una connessione in MySQL Workbench per connettersi al database con le credenziali dell'amministratore.

Fase 1: creazione di un utente del database Amazon RDS

Innanzitutto, è necessario un utente le cui credenziali saranno memorizzate nel segreto. Per creare l'utente, accedi al database Amazon RDS con le credenziali di amministratore archiviate in un segreto. Per semplicità, nel tutorial, si crea un utente con piena autorizzazione per un database. In un ambiente di produzione questo non succede normalmente e ti consigliamo di seguire il principio del privilegio minimo.

Recuperare la password dell'amministratore

1. Nella console Amazon RDS scegli il database.
2. Nella scheda Configuration (Configurazione), in Master Credentials ARN (ARN delle credenziali principali), scegli Manage in Secrets Manager (gestisci in secrets manager).

   Si apre la console Secrets Manager.
3. Nella pagina dei dettagli del segreto, scegli Retrieve secret value (Recupera il valore di un segreto).
4. La password viene visualizzata nella sezione Secret value (valore segreto).

Per creare un utente del database

1. In MySQL Workbench, fai clic con il pulsante destro del mouse sulla SecretsManagerTutorialconnessione e quindi scegli Modifica connessione.
2. Nella finestra di dialogo Manage Server Connections (Gestisci connessioni al server), per Username (Nome utente), inserisci **`admin`** e scegli Close (Chiudi).
3. Tornando in MySQL Workbench, scegli la connessione. SecretsManagerTutorial

4. Inserisci la password dell'amministratore che hai recuperato dal segreto.
5. In MySQL Workbench, nella finestra Query, inserisci i seguenti comandi (inclusa una password sicura) e quindi scegli Execute (Esegui). La funzione di rotazione verifica il segreto aggiornato utilizzando SELECT, quindi **dbuser** deve avere almeno quel privilegio.

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

   Nella finestra Output, viene visualizzato l'esito positivo dei comandi.

Fase 2: creazione di un segreto per le credenziali utente del database

Successivamente, creerai un segreto per archiviare le credenziali dell'utente appena creato e attiverai la rotazione automatica, inclusa una rotazione immediata. Secrets Manager ruota il segreto, il che significa che la password viene generata programmaticamente: nessun essere umano ha visto questa nuova password. L'avvio immediato della rotazione può anche aiutarti a determinare se la rotazione è impostata correttamente.

1. Apri la console Secrets Manager all'indirizzo https://console.aws.amazon.com/secretsmanager/.
2. Scegli Archivia un nuovo segreto.
3. Nella pagina Choose secret type (Scegli il tipo di segreto), effettua le seguenti operazioni:
   a. Per Secret type (Tipo segreto), scegli Credentials for Amazon RDS database (Credenziali per il database Amazon RDS).
   b. Per Credentials (Credenziali), inserisci il nome utente **dbuser** e la password inseritaper l'utente del database creato utilizzando MySQL Workbench.
   c. Per Database, scegli secretsmanagertutorialdb.
   d. Scegli Next (Successivo).
4. Nella pagina Configure secret (Configura il segreto), per Secret name (Nome del segreto), inserisci **SecretsManagerTutorialDbuser** e scegli Next (Successivo).
5. Nella pagina Configure rotation (Configura la rotazione), effetua le seguenti operazioni:
   a. Attiva Automatic rotation (Rotazione automatica).
   b. Per Rotation schedule (Pianificazione della rotazione), imposta una pianificazione di Days (Giorni): **2** Days with Duration: **2h** (2 giorni con durata: 2 h). Mantieni selezionato Rotate immediately (Ruota immediatamente).

c. Per Rotation function (Funzione di rotazione), scegli Create a rotation function (Crea una funzione di rotazione) e per il nome della funzione inserisci **`tutorial-single-user-rotation`**.

d. Per Strategia di rotazione scegli Utente singolo.

e. Scegli Next (Successivo).

6. Nella pagina Review (Rivedi), scegli Store (Archivia).

   Secrets Manager torna alla pagina dei dettagli segreti. Nella parte superiore della pagina, è possibile visualizzare lo stato della configurazione di rotazione. Secrets Manager utilizza CloudFormation per creare risorse come la funzione di rotazione Lambda e un ruolo di esecuzione che esegue la funzione Lambda. Al CloudFormation termine, il banner diventa Segreto programmato per la rotazione. La prima rotazione è completa.

Fase 3: esecuzione del test della password ruotata

Dopo la prima rotazione segreta, che potrebbe richiedere alcuni secondi, è possibile verificare che il segreto contenga ancora credenziali valide. La password nel segreto è cambiata rispetto alle credenziali originali.

Recuperare la nuova password dal segreto

1. Apri la console Secrets Manager all'indirizzo https://console.aws.amazon.com/secretsmanager/.
2. Scegli Secrets (Segreti) e quindi scegli il segreto **`SecretsManagerTutorialDbuser`**.
3. Nella pagina Secret details (Dettagli del segreto), scorri e scegli Retrieve secret value (Recupera il valore del segreto).
4. Nella tabella Key/value (Chiave/valore), copia il Secret value (Valore segreto) per la **`password`**.

Testare le credenziali

1. In MySQL Workbench, fai clic con il pulsante destro del mouse sulla SecretsManagerTutorialconnessione e quindi scegli Modifica connessione.
2. Nella finestra di dialogo Manage Server Connections (Gestisci connessioni al server), per Username (Nome utente), inserisci **`dbuser`** e scegli Close (Chiudi).
3. Tornando in MySQL Workbench, scegli la connessione. SecretsManagerTutorial

4. Nella finestra di dialogo Open SSH Connection (Apri connessione SSH), per Password incolla la password recuperata dal segreto, quindi scegli OK.

   Se le credenziali sono valide, MySQL Workbench si apre alla pagina di progettazione del database.

Fase 4: Eliminazione delle risorse

Per evitare potenziali addebiti, elimina il segreto creato in questo tutorial. Per istruzioni, consulta the section called "Eliminare un segreto".

Per ripulire le risorse create nel tutorial precedente, consulta the section called "Fase 4: Eliminazione delle risorse".

Fasi successive

- Ottieni informazioni su come recuperare i segreti nelle applicazioni. Per informazioni, consulta Ottieni segreti.
- Scopri altri programmi di rotazione. Per informazioni, consulta the section called "Pianificazioni di rotazione".

Crea un Gestione dei segreti AWS segreto

Un segreto può essere una password, un set di credenziali come nome utente e password, un OAuth token o altre informazioni segrete archiviate in forma crittografata in Secrets Manager.

> **Tip**
>
> Per le credenziali utente amministratore di Amazon RDS e Amazon Redshift, ti consigliamo di utilizzare i segreti gestiti. Puoi creare il segreto gestito tramite il servizio di gestione, quindi puoi utilizzare la rotazione gestita.

Quando si utilizza la console per archiviare le credenziali del database per un database di origine replicato in altre regioni, il segreto contiene le informazioni di connessione per il database di origine. Se poi replichi il segreto, le repliche saranno copie del segreto di origine e conterranno le stesse informazioni di connessione. È possibile aggiungere altre key/value coppie al segreto per le informazioni di connessione regionali.

Per creare un segreto, sono necessarie le autorizzazioni concesse dalla politica SecretsManagerReadWrite gestita.

Secrets Manager genera una voce di CloudTrail registro quando si crea un segreto. Per ulteriori informazioni, consulta the section called "Accedi con AWS CloudTrail ".

Come creare un segreto (console)

1. Apri la console Secrets Manager all'indirizzo https://console.aws.amazon.com/secretsmanager/.
2. Scegli Archivia un nuovo segreto.
3. Nella pagina Choose secret type (Scegli il tipo di segreto), effettua le seguenti operazioni:

    a. Per Tipo di segreto, procedere in uno dei seguenti modi:

    - Per memorizzare le credenziali del database, scegli il tipo di credenziali del database da archiviare. Quindi scegli il database e inserisci le credenziali.
    - Per memorizzare le chiavi API, i token di accesso e le credenziali che non sono per i database, scegli Altro tipo di segreto.

In Coppie chiave/valore, inserisci il segreto sotto forma di coppie Chiave/valore JSON oppure scegli la scheda Testo normale e inserisci il segreto in qualsiasi formato. Puoi archiviare fino a 65536 byte nel segreto. Alcuni esempi:

API key

Inserisci come key/value coppie:

`ClientID` : *`my_client_id`*

`ClientSecret` : *`wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY`*

OAuth token

Inserisci come testo semplice:

`AKIAI44QH8DHBEXAMPLE`

Digital certificate

Inserisci come testo semplice:

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

Private key

Inserisci come testo semplice:

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

- Per archiviare un segreto esterno gestito da un partner Secrets Manager, scegli Partner secret. Quindi scegli il partner e fornisci i dettagli che identificano il segreto del partner. Per informazioni dettagliate, vedi Utilizzo di segreti esterni Gestione dei segreti AWS gestiti per gestire segreti di terze parti.

b. Per la chiave di crittografia, scegli AWS KMS key quella utilizzata da Secrets Manager per crittografare il valore segreto. Per ulteriori informazioni, consulta Crittografia e decrittografia del segreto.

- Nella maggior parte dei casi, scegli aws/secretsmanager per usare for Secrets Chiave gestita da AWS Manager. L'utilizzo di questa chiave non prevede costi aggiuntivi.
- Se devi accedere al segreto da un altro Account AWS o se desideri utilizzare la tua chiave KMS in modo da poterla ruotare o applicare una politica chiave, scegli una chiave gestita dal cliente dall'elenco o scegli Aggiungi nuova chiave per crearne una. Per informazioni sui costi di utilizzo di una chiave gestita dal cliente, consulta la sezione Prezzi.

  È necessario avere le the section called "Autorizzazioni per la chiave KMS". Per informazioni sull'accesso tra account, consulta the section called "Accesso multi-account".

c. Scegli Next (Successivo).

4. Nella pagina Configure secret (Configura il segreto), effettua le seguenti operazioni:

   a. Inserisci un Secret name (Nome del segreto) e una Description (Descrizione) descrittivi. I nomi segreti possono contenere da 1 a 512 caratteri alfanumerici e /_+ =.@-.

   b. (Facoltativo) Se hai creato un segreto esterno, inserisci i metadati richiesti dal partner Secrets Manager che detiene il segreto.

   c. (Facoltativo) Nella sezione Tags (Tag) aggiungere tag al segreto. Per le strategie di assegnazione dei tag, vedere the section called "Tag segreti ". Non archiviare informazioni sensibili nei tag perché non sono crittografate.

   d. (Facoltativo) In Permessi delle risorse, per aggiungere una policy delle risorse al tuo segreto, scegli Modifica delle autorizzazioni. Per ulteriori informazioni, consulta the section called "Policy basate sulle risorse".

   e. (Facoltativo) In Replica segreto, per replicare il tuo segreto su un altro Regione AWS, scegli Replica segreto. Puoi replicare il tuo segreto immediatamente o tornare e replicarlo in un secondo momento. Per ulteriori informazioni, consulta Replica in più regioni.

   f. Scegli Next (Successivo).

5. (Facoltativo) Nella pagina Configure rotation (Configura la rotazione), puoi attivare la rotazione automatica. Puoi anche disattivare la rotazione e poi riattivarla in un secondo momento. Per ulteriori informazioni, consulta Rotazione dei segreti . Scegli Next (Successivo).

6. Nella pagina Review (Revisione), rivedi i dettagli dei segreti e quindi scegli Store (Archivia).

   Secrets Manager ritorna all'elenco dei segreti. Se il segreto nuovo non viene visualizzato, scegli il pulsante aggiorna.

AWS CLI

Quando immetti i comandi in una shell dei comandi, c'è il rischio che la cronologia dei comandi sia accessibile o che le utilità abbiano accesso ai parametri dei comandi. Per informazioni, consulta the section called "Riduci i rischi derivanti dall'utilizzo di per archiviare i tuoi AWS CLI segreti Gestione dei segreti AWS".

Example Crea un segreto dalle credenziali del database in un file JSON

L'esempio di `create-secret` seguente mostra come creare un segreto partendo dalle credenziali in un file. Per ulteriori informazioni, consulta Caricamento AWS CLI dei parametri da un file nella Guida per l' AWS CLI utente.

Affinché Secrets Manager sia in grado di ruotare il segreto, devi assicurarti che il JSON corrisponda alla Struttura JSON di un segreto .

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

Contenuti di mycreds.json:

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

Example Creazione di un segreto

L'esempio di `create-secret` seguente mostra come creare un segreto con due coppie chiave-valore.

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

Example Creazione di un segreto

L'`create-secret`esempio seguente crea un segreto con due tag.

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

AWS SDK

Per creare un segreto utilizzando uno dei AWS SDKs, usa l'`CreateSecret`azione. Per ulteriori informazioni, consulta the section called "AWS SDKs".

Cosa c'è in un segreto di Secrets Manager?

In Secrets Manager, un segreto è costituito dalle informazioni del segreto, dal valore del segreto e dai metadati sul segreto. Un valore segreto può essere di tipo stringa o binario.

Per memorizzare più valori di stringa in un unico segreto, ti consigliamo di utilizzare una stringa di testo JSON con coppie chiave-valore, ad esempio:

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

Per quanto riguarda i segreti del database, se desideri attivare la rotazione automatica, il segreto deve contenere informazioni di connessione per il database nella struttura JSON corretta. Per ulteriori informazioni, consulta the section called "Struttura JSON di un segreto ".

Metadati

I metadati di un segreto includono:

- Amazon Resource Name (ARN) ha il seguente formato:

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

  Secrets Manager include sei caratteri casuali alla fine del nome del segreto per garantire che l'ARN del segreto sia univoco. Se il segreto originale viene eliminato e quindi viene creato un nuovo segreto con lo stesso nome, i due segreti sono diversi a ARNs causa di questi caratteri. Gli utenti con accesso al vecchio segreto non ottengono automaticamente l'accesso al nuovo segreto perché ARNs sono diversi.
- Il nome del segreto, una descrizione, una policy di risorse e i tag.
- L'ARN per una chiave di crittografia e AWS KMS key che Secrets Manager utilizza per crittografare e decrittografare il valore segreto. Secrets Manager archivia sempre il testo del segreto in un modulo crittografato e crittografa sempre il segreto in transito. Per informazioni, consulta the section called "Crittografia e decrittografia del segreto".
- Informazioni su come ruotare il segreto, se si imposta la rotazione. Per informazioni, consulta Rotazione dei segreti .

Secrets Manager utilizza le policy di autorizzazione IAM per garantire che solo gli utenti autorizzati possano accedere o modificare un segreto. Per informazioni, consulta Autenticazione e controllo degli accessi per Gestione dei segreti AWS.

Un segreto ha versioni che contengono copie del valore segreto crittografato. Quando si cambia il valore del segreto o il segreto viene ruotato, Secrets Manager crea una nuova versione. Per informazioni, consulta the section called "Versioni segrete".

È possibile utilizzare un segreto su più Regioni AWS siti replicandolo. Quando si replica un segreto, si crea una copia dell'originale o segreto primario chiamato un Segreto di replica. Il segreto di replica rimane collegato al segreto primario. Per informazioni, consulta Replica in più regioni.

Per informazioni, consulta Gestisci i segreti.

Versioni segrete

Un segreto ha versioni che contengono copie del valore segreto crittografato. Quando si cambia il valore del segreto o il segreto viene ruotato, Secrets Manager crea una nuova versione.

Secrets Manager non memorizza la cronologia lineare dei segreti con le rispettive versioni. Al contrario, tiene traccia di tre versioni specifiche etichettandole:

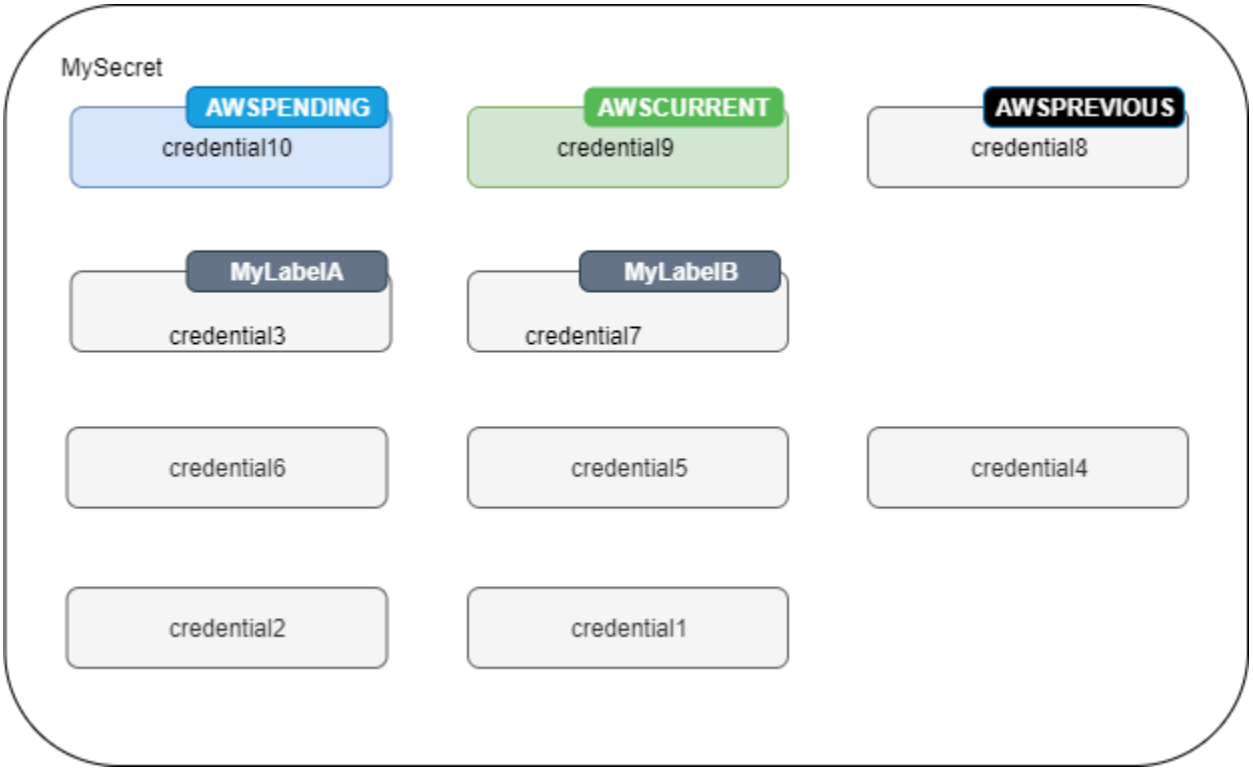
- La versione attuale: `AWSCURRENT`
- La versione precedente — `AWSPREVIOUS`
- La versione in sospeso (durante la rotazione) — `AWSPENDING`

Un segreto ha sempre una versione etichettata `AWSCURRENT`; per impostazione predefinita, quando recuperi il valore del segreto, Secrets Manager restituisce tale versione.

Puoi anche etichettare le versioni con le tue etichette [`update-secret-version-stage`]chiamando il AWS CLI. È possibile assegnare alle versioni di un segreto fino a 20 etichette. Due versioni di un segreto non possono avere la stessa etichetta di gestione temporanea. Le versioni possono avere più etichette.

Secrets Manager non rimuove mai le versioni etichettate, ma le versioni senza etichetta sono considerate obsolete. Secrets Manager rimuove le versioni obsolete quando il loro numero diventa maggiore di 100. Secrets Manager non rimuove le versioni create meno di 24 ore prima.

La figura seguente mostra un segreto con versioni AWS etichettate e versioni etichettate dal cliente. Le versioni senza etichette sono considerate obsolete e verranno rimosse da Secrets Manager in un momento futuro.



Struttura dei segreti JSON Gestione dei segreti AWS

È possibile memorizzare qualsiasi testo o file binario in un segreto di Secrets Manager fino alla dimensione massima di 65.536 byte.

Se si utilizzathe section called "Rotazione tramite funzione Lambda", un segreto deve contenere campi JSON specifici previsti dalla funzione di rotazione. Ad esempio, per un segreto che contiene le credenziali del database, la funzione di rotazione si connette al database per aggiornare le credenziali, quindi il segreto deve contenere le informazioni di connessione al database.

Se si utilizza la console per modificare la rotazione di un segreto del database, il segreto deve contenere coppie chiave-valore JSON specifiche che identificano il database. Secrets Manager utilizza questi campi per interrogare il database per trovare il VPC corretto in cui memorizzare una funzione di rotazione.

I nomi delle chiavi JSON fanno distinzione tra maiuscole e minuscole.

Argomenti

- Credenziali Amazon RDS e Aurora
- Credenziali Amazon Redshift
- Credenziali Serverless Amazon Redshift
- Credenziali Amazon DocumentDB
- Amazon Timestream per la struttura segreta di InfluxDB
- ElastiCache Credenziali Amazon
- Credenziali Active Directory

Credenziali Amazon RDS e Aurora

Per utilizzare i modelli di funzioni di rotazione forniti da Secrets Manager, utilizzare la seguente struttura JSON. È possibile aggiungere altre key/value coppie, ad esempio per contenere informazioni di connessione per database di replica in altre regioni.

DB2

> Poiché gli utenti non possono modificare le proprie password, per le istanze Amazon RDS Db2 è necessario fornire le credenziali di amministratore in un segreto separato.

```
{
```

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

MariaDB

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

MySQL

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

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

Oracle

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

Postgres

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

SQLServer

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

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

Credenziali Amazon Redshift

Per utilizzare i modelli di funzioni di rotazione forniti da Secrets Manager, utilizzare la seguente struttura JSON. È possibile aggiungere altre key/value coppie, ad esempio per contenere informazioni di connessione per database di replica in altre regioni.

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

Credenziali Serverless Amazon Redshift

Per utilizzare i modelli di funzioni di rotazione forniti da Secrets Manager, utilizzare la seguente struttura JSON. È possibile aggiungere altre key/value coppie, ad esempio per contenere informazioni di connessione per database di replica in altre regioni.

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

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

Credenziali Amazon DocumentDB

Per utilizzare i modelli di funzioni di rotazione forniti da Secrets Manager, utilizzare la seguente struttura JSON. È possibile aggiungere altre key/value coppie, ad esempio per contenere informazioni di connessione per database di replica in altre regioni.

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

Amazon Timestream per la struttura segreta di InfluxDB

Per ruotare i segreti di Timestream, puoi utilizzare i modelli di rotazione. the section called "Amazon Timestream per InfluxDB"

Per ulteriori informazioni, consulta l'articolo Come Amazon Timestream for InfluxDB utilizza i segreti nella Amazon Timestream Developer Guide.

I segreti di Timestream devono essere nella struttura JSON corretta per poter utilizzare i modelli di rotazione. Per ulteriori informazioni, consulta What's in the secret nella Amazon Timestream Developer Guide.

ElastiCache Credenziali Amazon

L'esempio seguente mostra la struttura JSON di un segreto che ElastiCache memorizza le credenziali.

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

Per ulteriori informazioni, consulta Rotazione automatica delle password per gli utenti nella Amazon ElastiCache User Guide.

Credenziali Active Directory

AWS Directory Service utilizza segreti per archiviare le credenziali di Active Directory. Per ulteriori informazioni, consulta Unire senza problemi un'istanza Amazon EC2 Linux alla tua Managed AD Active Directory nella Guida AWS Directory Service all'amministrazione. Seamless Domain Join richiede i nomi chiave riportati negli esempi seguenti. Se non utilizzi Seamless Domain Join, puoi modificare i nomi delle chiavi del segreto utilizzando le variabili di ambiente come descritto nel codice del modello della funzione di rotazione.

Per ruotare i segreti di Active Directory, puoi utilizzare i modelli di rotazione di Active Directory.

Active Directory credential

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

Se desideri ruotare il segreto, includi l'ID della directory del dominio.

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

Se il segreto viene utilizzato insieme a un segreto che contiene un keytab, includi il segreto keytab. ARNs

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

Active Directory keytab

Per informazioni sull'utilizzo dei file keytab per l'autenticazione su account Active Directory su Amazon EC2, consulta Distribuzione e configurazione dell'autenticazione Active Directory con SQL Server 2017 su 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
}
```

Gestisci i segreti con Gestione dei segreti AWS

Argomenti

Aggiorna il valore di un Gestione dei segreti AWS segreto

Per aggiornare il valore del tuo segreto, puoi utilizzare la console, la CLI o un SDK. Quando aggiorni il valore del segreto, Secrets Manager crea una nuova versione del segreto con l'etichetta temporanea `AWSCURRENT`. Puoi ancora accedere alla versione precedente con l'etichetta `AWSPREVIOUS`. Puoi anche aggiungere le tue etichette. Per ulteriori informazioni, consulta Controllo delle versioni di Secrets Manager.

Per aggiornare il valore segreto (console)

1. Apri la console Secrets Manager all'indirizzo https://console.aws.amazon.com/secretsmanager/.
2. Dall'elenco dei segreti, scegli il segreto.
3. Nella pagina dei dettagli del segreto, nella sezione Panoramica in Valore del segreto, scegli Recupera il valore del segreto, quindi scegli Modifica.

AWS CLI

Per aggiornare il valore del segreto (AWS CLI)

- Quando immetti i comandi in una shell dei comandi, c'è il rischio che la cronologia dei comandi sia accessibile o che le utilità abbiano accesso ai parametri dei comandi. Per informazioni, consulta the section called "Riduci i rischi derivanti dall'utilizzo di per archiviare i tuoi AWS CLI segreti Gestione dei segreti AWS".

  L'esempio di `put-secret-value` seguente mostra come creare una nuova versione di un segreto con due coppie chiave-valore.

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

  L'esempio di `put-secret-value` mostra come creare una nuova versione di un'etichetta temporanea personalizzata. La nuova versione avrà le etichette `MyLabel` e `AWSCURRENT`.

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

AWS SDK

Ti consigliamo di evitare di chiamare `PutSecretValue` o `UpdateSecret` ad una frequenza sostenuta di più di una volta ogni 10 minuti. Quando chiami `PutSecretValue` o `UpdateSecret` per aggiornare il valore del segreto, Secrets Manager crea una nuova versione del segreto. Secrets Manager rimuove le versioni obsolete quando sono più di 100, ma non rimuove le versioni create da meno di 24 ore. Se aggiorni il valore segreto più di una volta ogni 10 minuti, crei più versioni di quelle che Secrets Manager rimuove e raggiungerai la quota massima prevista per le versioni di un segreto.

Per aggiornare un valore segreto, utilizza le seguenti azioni: `UpdateSecret` o `PutSecretValue`. Per ulteriori informazioni, consulta the section called "AWS SDKs".

Genera una password con Secrets Manager

Uno schema comune per l'utilizzo di Secrets Manager consiste nel generare una password in Secrets Manager e quindi utilizzarla nel database o nel servizio. Puoi farlo utilizzando i seguenti metodi:

- CloudFormation — VediCloudFormation.
- AWS CLI — Vedi `get-random-password`.
- AWS SDKs — Vedi `GetRandomPassword`.

Ripristina un segreto a una versione precedente

Puoi ripristinare un segreto a una versione precedente spostando le etichette allegate alle versioni segrete utilizzando il AWS CLI. Per informazioni su come Secrets Manager archivia le versioni dei segreti, vederethe section called "Versioni segrete".

L'`update-secret-version-stage`esempio seguente sposta AWSCURRENT l'etichetta temporanea alla versione precedente di un segreto, che ripristina il segreto alla versione precedente. Per trovare l'ID della versione precedente, usa `list-secret-version-ids`o visualizza le versioni nella console Secrets Manager.

Per questo esempio, la versione con l'etichetta è a1b2c3d4-5678-90ab-cdef- e la versione con l' AWSCURRENT etichetta è a1b2c3d4-5678-90ab-cdef-. EXAMPLE11111 AWSPREVIOUS EXAMPLE22222 In questo esempio, si sposta l'etichetta dalla versione 11111 alla 22222. AWSCURRENT Poiché l' AWSCURRENT etichetta viene rimossa da una versione, sposta `update-secret-version-stage` automaticamente l' AWSPREVIOUS etichetta a quella versione (11111). L'effetto è che le AWSPREVIOUS versioni AWSCURRENT e vengono scambiate.

```
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
```

Modificare la chiave di crittografia per un Gestione dei segreti AWS segreto

Secrets Manager utilizza la crittografia a busta con AWS KMS chiavi e chiavi dati per proteggere ogni valore segreto. Per ogni segreto, puoi scegliere quale chiave KMS usare. È possibile utilizzare o utilizzare una chiave gestita dal cliente. Chiave gestita da AWS aws/secretsmanager Nella maggior parte dei casi, si consiglia di aws/secretsmanager utilizzarlo e non è previsto alcun costo per l'utilizzo. Se devi accedere al segreto da un'altra Account AWS persona o se desideri utilizzare la tua chiave KMS in modo da poterla ruotare o applicare una politica di chiave, usa una. chiave gestita dal clienteÈ necessario avere le the section called "Autorizzazioni per la chiave KMS". Per informazioni sui costi di utilizzo di una chiave gestita dal cliente, consulta la sezione Prezzi.

Puoi modificare la chiave di crittografia per il segreto. Ad esempio, se desideri accedere al segreto da un altro account e il segreto è attualmente crittografato utilizzando la chiave AWS gestita`aws/secretsmanager`, puoi passare a un. chiave gestita dal cliente

> **Tip**
>
> Se desideri ruotare il tuo chiave gestita dal cliente, ti consigliamo di utilizzare la rotazione AWS KMS automatica dei tasti. Per ulteriori informazioni, consulta Rotazione AWS KMS dei tasti.

Quando si modifica la chiave di crittografia, Secrets Manager cripta nuovamente `AWSCURRENT` e `AWSPENDING` le `AWSPREVIOUS` versioni con la nuova chiave. Per evitare di nasconderti il segreto, Secrets Manager mantiene tutte le versioni esistenti crittografate con la chiave precedente. Ciò significa che è possibile decrittografare `AWSCURRENT` `AWSPENDING` le `AWSPREVIOUS` versioni con la chiave precedente o quella nuova. Se non si dispone dell'`kms:Decrypt`autorizzazione per la chiave precedente, quando si modifica la chiave di crittografia, Secrets Manager non può decrittografare le versioni segrete per crittografarle nuovamente. In questo caso, le versioni esistenti non vengono ricrittografate.

Per fare in modo che `AWSCURRENT` possa essere decifrata solo con la nuova chiave di crittografia, crea una nuova versione del segreto con la nuova chiave. Quindi, per poter decifrare la versione `AWSCURRENT` segreta, devi avere l'autorizzazione per la nuova chiave.

Se disattivi la chiave di crittografia precedente, non potrai decrittografare nessuna versione segreta ad eccezione di `AWSCURRENT`, `AWSPENDING` e `AWSPREVIOUS`. Se disponi di altre versioni segrete

etichettate per le quali desideri mantenere l'accesso, devi ricreare tali versioni con la nuova chiave di crittografia utilizzando the section called "AWS CLI".

Per modificare la chiave di crittografia per un segreto (console)

1. Apri la console Secrets Manager all'indirizzo https://console.aws.amazon.com/secretsmanager/.
2. Dall'elenco dei segreti, scegli il segreto.
3. Nella pagina dei dettagli del segreto, nella sezione Secrets details (Dettagli segreti), scegli Actions (Operazioni), quindi scegli Edit encryption key (Modifica chiave di crittografia).

AWS CLI

Se modifichi la chiave di crittografia per un segreto e disattivi la chiave di crittografia precedente, non sarà possibile decrittografare nessuna versione segreta ad eccezione di `AWSCURRENT`, `AWSPENDING` e `AWSPREVIOUS`. Se disponi di altre versioni segrete etichettate per le quali desideri mantenere l'accesso, devi ricreare tali versioni con la nuova chiave di crittografia utilizzando the section called "AWS CLI".

Per modificare la chiave di crittografia per un segreto (AWS CLI)

1. L'esempio di `update-secret` seguente mostra come aggiornare la chiave KMS utilizzata per crittografare il valore del segreto. La chiave KMS deve trovarsi nella stessa Regione del segreto.

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

2. (Facoltativo) Se disponi di versioni segrete con etichette personalizzate, per poterle crittografare nuovamente utilizzando la nuova chiave è necessario ricrearle.

   Quando immetti i comandi in una shell dei comandi, c'è il rischio che la cronologia dei comandi sia accessibile o che le utilità abbiano accesso ai parametri dei comandi. Per informazioni, consulta the section called "Riduci i rischi derivanti dall'utilizzo di per archiviare i tuoi AWS CLI segreti Gestione dei segreti AWS".

   a. Ottieni il valore della versione segreta.

```
aws secretsmanager get-secret-value \
```

```
      --secret-id MyTestSecret \
      --version-stage MyCustomLabel
```

Annota il valore segreto.

b. Crea una nuova versione con quel valore.

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

Modifica un Gestione dei segreti AWS segreto

Puoi modificare i metadati di un segreto dopo che è stato creato in base a chi ha creato il segreto. Per i segreti creati da altri servizi, potrebbe essere necessario utilizzare l'altro servizio per aggiornarlo o ruotarlo.

Per determinare chi gestisce un segreto, puoi rivedere il nome del segreto. I segreti gestiti da altri servizi sono preceduti dall'ID di quel servizio. Oppure, chiama describe-secret AWS CLI, quindi esamina il campo. `OwningService` Per ulteriori informazioni, consulta Segreti gestiti da altri servizi.

Per i segreti da te gestiti, puoi modificare la descrizione, la policy basata sulle risorse, la chiave di crittografia e i tag. Puoi anche modificare il valore delle informazioni crittografate del segreto, sebbene sia consigliabile utilizzare la rotazione per aggiornare i valori del segreto che contengono credenziali. La rotazione aggiorna sia il segreto in Secrets Manager che le credenziali del database o del servizio. Questo consente di mantenere i segreti sincronizzati automaticamente in modo che quando i client richiedono un valore del segreto, recuperano sempre un set di credenziali funzionante. Per ulteriori informazioni, consulta Rotazione dei segreti .

Secrets Manager genera una voce di CloudTrail registro quando si modifica un segreto. Per ulteriori informazioni, consulta the section called “Accedi con AWS CloudTrail ”.

Aggiornamento di un segreto da te gestito (console)

1. Apri la console Secrets Manager all'indirizzo https://console.aws.amazon.com/secretsmanager/.
2. Dall'elenco dei segreti, scegli il segreto.
3. Nella pagina dei dettagli del segreto, completa una delle seguenti operazioni:

Tieni presente che non puoi modificare il nome o l'ARN di un segreto.

- Per aggiornare la descrizione, nella sezione Secrets details (Dettagli segreti) scegli Actions (Operazioni), quindi scegli Edit description (Modifica descrizione).
- Per aggiornare la chiave di crittografia, consulta the section called "Modifica la chiave di crittografia per un segreto".
- Per aggiornare i tag, nella sezione Tag, scegli Modifica tag. Per informazioni, consulta the section called "Tag segreti ".
- Per aggiornare il valore del segreto, consulta the section called "Aggiorna un valore segreto".
- Per aggiornare le autorizzazioni per il segreto, nella sezione Panoramica scegli Modifica autorizzazioni. Per informazioni, consulta the section called "Policy basate sulle risorse".
- Per aggiornare la rotazione per il segreto, nella sezione Rotazione scegli Modifica rotazione. Per informazioni, consulta Rotazione dei segreti .
- Per replicare il tuo segreto in altre regioni, consulta Replica in più regioni.
- Se il tuo segreto contiene repliche, puoi modificare la chiave di crittografia per una replica. Nella sezione Replica, seleziona il pulsante di opzione per la replica e quindi dal menu Operazioni, scegli Modifica la chiave di crittografia. Per informazioni, consulta the section called "Crittografia e decrittografia del segreto".
- Per modificare un segreto in modo che sia gestito da un altro servizio, è necessario ricrearlo in tale servizio. Per informazioni, consulta Segreti gestiti da altri servizi.

AWS CLI

Example Aggiornamento della descrizione di un segreto

L'esempio di `update-secret` seguente mostra come aggiornare la descrizione di un segreto.

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

AWS SDK

Ti consigliamo di evitare di chiamare `PutSecretValue` o `UpdateSecret` ad una frequenza sostenuta di più di una volta ogni 10 minuti. Quando chiami `PutSecretValue` o `UpdateSecret`

per aggiornare il valore del segreto, Secrets Manager crea una nuova versione del segreto. Secrets Manager rimuove le versioni obsolete quando sono più di 100, ma non rimuove le versioni create da meno di 24 ore. Se aggiorni il valore segreto più di una volta ogni 10 minuti, crei più versioni di quelle che Secrets Manager rimuove e raggiungerai la quota massima prevista per le versioni di un segreto.

Per aggiornare un segreto, utilizza le seguenti azioni: `UpdateSecret` o `ReplicateSecretToRegions`. Per ulteriori informazioni, consulta the section called "AWS SDKs".

Trova segreti in Gestione dei segreti AWS

Quando cerchi segreti senza filtro, Secrets Manager fa corrispondere le parole chiave nel nome del segreto, nella descrizione, nella chiave tag e nel valore del tag. La ricerca senza filtri non fa distinzione tra maiuscole e minuscole e ignora caratteri speciali, come spazio, /, _, =, # e utilizza solo numeri e lettere. Quando esegui una ricerca senza filtri, Secrets Manager analizza la stringa di ricerca per convertirla in parole separate. Le parole sono separate da qualsiasi modifica da maiuscola a minuscola, da lettera a numero o da a punteggiatura. number/letter Ad esempio, inserendo il termine di ricerca `credsDatabase#892`, viene effettuata la ricerca di `creds`, `Database` e `892` in nome, descrizione e chiave e valore di tag.

Secrets Manager genera una voce di CloudTrail registro quando si elencano i segreti. Per ulteriori informazioni, consulta the section called "Accedi con AWS CloudTrail ".

Secrets Manager è un servizio regionale e restituisce solo i segreti entro la regione selezionata.

Filtri di ricerca

Se non si utilizza alcun filtro, Secrets Manager suddivide la stringa di ricerca in parole e quindi cerca le corrispondenze in tutti gli attributi. Questa ricerca non fa distinzione tra maiuscole e minuscole. Ad esempio, la ricerca di segreti **`My_Secret`** corrisponde alla parola my o secret nel nome, nella descrizione o nei tag.

Puoi applicare i seguenti filtri alla ricerca:

Name

> Corrisponde all'inizio dei nomi dei segreti; distinzione tra maiuscole e minuscole. Ad esempio: Nome: **`Data`** restituisce un segreto chiamato `DatabaseSecret`, ma non `databaseSecret` o `MyData`.

Description

Corrisponde alle parole nelle descrizioni segrete, nessuna distinzione tra lettere maiuscole e minuscole. Ad esempio:Descrizione: **`My Description`** abbina i segreti con le seguenti descrizioni:

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

Gestito da

Trova segreti gestiti da servizi esterni AWS, ad esempio:

- 1 password
- Senza chiavi
- CyberArk
- HashiCorp

Servizio di proprietà

Corrisponde all'inizio dei prefissi ID del servizio, nessuna distinzione tra lettere maiuscole e minuscole. Ad esempio, **`my-ser`** abbina i segreti gestiti dai servizi al prefisso `my-serv` e `my-service`. Per ulteriori informazioni, consulta Segreti gestiti da altri servizi.

Segreti replicati

Puoi filtrare i segreti principali, di replica o quelli che non vengono replicati.

Tasti Tag

Corrisponde all'inizio dei tasti tag; distinzione tra maiuscole e minuscole. Ad esempio: Tasto tag: **`Prod`** restituisce segreti con il tag `Production` e `Prod1`, ma non segreti con il tag `prod` o `1 Prod`.

Valori Tag

Corrisponde all'inizio dei valori dei tag; distinzione tra maiuscole e minuscole. Ad esempio: Valore tag: **`Prod`** restituisce segreti con il tag `Production` e `Prod1`, ma non segreti con il valore del tag `prod` o `1 Prod`.

AWS CLI

Example Elencazione dei segreti nell'account

L'esempio di `list-secrets` seguente mostra come ottenere un elenco dei segreti del proprio account.

```
aws secretsmanager list-secrets
```

Example Filtraggio dell'elenco dei segreti nell'account

L'esempio di `list-secrets` seguente mostra come ottenere un elenco dei segreti del proprio account che contengono Test nel nome. Il filtro per nome fa distinzione tra maiuscole e minuscole.

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

Example Scopri i segreti gestiti da altri servizi AWS

L'esempio `list-secrets` seguente ottiene un elenco di segreti gestiti da un servizio. Si specifica il servizio in base all'ID. Per ulteriori informazioni, consulta Segreti gestiti da altri servizi.

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

AWS SDK

Per trovare segreti usando uno di questi AWS SDKs, usa `ListSecrets`. Per ulteriori informazioni, consulta the section called "AWS SDKs".

Eliminare un Gestione dei segreti AWS segreto

A causa della natura critica dei segreti, rende Gestione dei segreti AWS intenzionalmente difficile l'eliminazione di un segreto. Secrets Manager non elimina immediatamente i segreti. Invece, Secrets Manager rende immediatamente inaccessibili i segreti e li pianifica per l'eliminazione dopo un intervallo di recupero di un minimo di sette giorni. Fino al termine dell'intervallo di recupero, puoi recuperare un segreto eliminato in precedenza. Non è previsto alcun costo per i segreti contrassegnati per l'eliminazione.

Non è possibile eliminare un segreto primario se viene replicato in altre regioni. Eliminare prima le repliche di lettura, poi eliminare il segreto primario. Quando si elimina una replica, questa viene eliminata immediatamente.

Non è possibile eliminare direttamente una versione di un segreto. Invece, rimuovi tutte le etichette di staging dalla versione utilizzando o SDK. AWS CLI AWS In questo modo, la versione viene contrassegnata come obsoleta e Secrets Manager può eliminarla automaticamente in background.

Se non sai se un'applicazione utilizza ancora un segreto, puoi creare un CloudWatch allarme Amazon per avvisarti di eventuali tentativi di accesso a un segreto durante la finestra di ripristino. Per ulteriori informazioni, consulta Monitora l'accesso ai Gestione dei segreti AWS segreti programmati per l'eliminazione.

Per eliminare un segreto, devi disporre di `secretsmanager:ListSecrets` e `secretsmanager:DeleteSecret`autorizzazioni.

Secrets Manager genera una voce di CloudTrail registro quando si elimina un segreto. Per ulteriori informazioni, consulta the section called "Accedi con AWS CloudTrail ".

Come aggiornare un segreto (console)

1. Apri la console Secrets Manager all'indirizzo https://console.aws.amazon.com/secretsmanager/.
2. Nell'elenco di segreti, scegli il segreto che desideri eliminare.
3. Nella sezione Secrets details (Dettagli Segreti) scegli Actions (Operazioni), quindi scegli Edit description (Modifica descrizione).
4. Nella finestra di dialogo Disabilitare l'eliminazione segreta e pianificare, in Periodo di attesa, inserisci il numero di giorni di attesa prima che l'eliminazione diventi definitiva. Secrets Manager collega un campo denominato `DeletionDate` e lo imposta sulla data e sull'ora correnti, a cui somma il numero di giorni specificati per l'intervallo di recupero.
5. Scegliere Schedule deletion (Pianifica eliminazione).

Come visualizzare i segreti eliminati

1. Apri la console Secrets Manager all'indirizzo https://console.aws.amazon.com/secretsmanager/.
2. Sulla pagina Segreti, seleziona Preferenze ().
3. Nella finestra di dialogo Preferences (Preferenze) seleziona Show secrets scheduled for deletion (Mostra segreti pianificati per la cancellazione) e quindi scegli Save (Salva).

Come eliminare un segreto di replica

1. Apri la console Secrets Manager all'indirizzo https://console.aws.amazon.com/secretsmanager/.
2. Scegli il segreto primario.
3. Nella sezione Replica di segreto, scegli il segreto di replica.
4. Dal menu Actions (Operazioni), scegliere Delete Replica (Elimina replica).

AWS CLI

Example Eliminare un segreto

L'esempio di `delete-secret` seguente mostra come eliminare un segreto. Puoi recuperare il segreto `restore-secret`entro la data e l'ora indicate nel campo di DeletionDate risposta. Per eliminare un segreto replicato in altre regioni, è necessario dapprima rimuovere le relative repliche con `remove-regions-from-replication`, quindi chiamare `delete-secret`.

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

Example Eliminazione immediata di un segreto

L'esempio di `delete-secret` seguente mostra come eliminare immediatamente il secreto senza un intervallo di recupero. Non è possibile recuperare questo segreto.

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

Example Eliminazione di un segreto di replica

L'esempio di `remove-regions-from-replication` seguente mostra come eliminare un segreto di replica nella Regione eu-west-3. Per eliminare un segreto primario replicato in altre Regioni, è necessario dapprima eliminare le relative repliche e poi chiamare `delete-secret`.

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

AWS SDK

Per eliminare un segreto, utilizza il comando `DeleteSecret`. Per eliminare la versione di un segreto, usa il comando `UpdateSecretVersionStage`. Per eliminare una replica, utilizza il comando `StopReplicationToReplica`. Per ulteriori informazioni, consulta the section called "AWS SDKs".

Ripristina un Gestione dei segreti AWS segreto

Secrets Manager considera un segreto pianificato per l'eliminazione obsoleto e non più accessibile direttamente. Una volta trascorso l'intervallo di recupero, Secrets Manager elimina il segreto definitivamente. Dopo che Secrets Manager ha eliminato il segreto, non è possibile recuperarlo. Prima della fine dell'intervallo di recupero, puoi recuperare il segreto e renderlo nuovamente accessibile. Il campo `DeletionDate` viene rimosso e l'eliminazione permanente pianificata viene annullata.

Per ripristinare un segreto e i metadati nella console devi disporre di `secretsmanager:ListSecrets` e `secretsmanager:RestoreSecret` autorizzazioni.

Secrets Manager genera una voce di CloudTrail registro quando si ripristina un segreto. Per ulteriori informazioni, consulta the section called "Accedi con AWS CloudTrail ".

Come ripristinare un segreto (console)

1. Apri la console Secrets Manager all'indirizzo https://console.aws.amazon.com/secretsmanager/.
2. Nell'elenco di segreti, scegli il segreto che desideri modificare.

   Se i segreti eliminati non vengono visualizzati nell'elenco dei segreti, scegli Preferenze ().
   Nella finestra di dialogo Preferences (Preferenze) seleziona Show secrets scheduled for deletion (Mostra segreti pianificati per l'eliminazione) e quindi scegli Save (Salva)
3. Nella sezione Secret details (Dettagli segreto) scegli Cancel deletion (Annulla eliminazione).
4. Nella finestra di dialogo Cancel secret deletion (Annulla eliminazione segreto) scegli Cancel deletion (Annulla eliminazione).

AWS CLI

Example Ripristino di un segreto precedentemente eliminato

L'esempio di `restore-secret` seguente mostra il ripristino di un segreto per il quale in precedenza era stata pianificata l'eliminazione.

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

AWS SDK

Per ripristinare un segreto contrassegnato per l'eliminazione, utilizzare il comando `RestoreSecret`. Per ulteriori informazioni, consulta the section called "AWS SDKs".

Taggare i segreti in Gestione dei segreti AWS

In Gestione dei segreti AWS, puoi assegnare metadati ai tuoi segreti utilizzando i tag. Un tag è una coppia chiave-valore che definisci per un segreto. I tag consentono di gestire AWS le risorse e organizzare i dati, incluse le informazioni di fatturazione.

Con i tag puoi:

- Gestire, cercare e filtrare segreti e altre risorse nel tuo AWS account
- Controlla l'accesso ai segreti in base ai tag allegati
- Tieni traccia e classifica le spese associate a segreti o progetti specifici

Per ulteriori informazioni sull'utilizzo dei tag per controllare l'accesso, consultathe section called "Controlla l'accesso ai segreti utilizzando i tag".

Per ulteriori informazioni sui tag di allocazione dei costi, vedere Utilizzo AWS dei tag di allocazione dei costi nella Guida per l' AWS Billing utente.

Per informazioni sulle quote dei tag e sulle restrizioni di denominazione, consulta Service quotas for tagging nella guida di riferimento generale.AWS I tag rispettano la distinzione tra maiuscole e minuscole.

Secrets Manager genera una voce di CloudTrail registro quando tagghi o rimuovi un tag da un segreto. Per ulteriori informazioni, consulta the section called "Accedi con AWS CloudTrail ".

> **Tip**
>
> Utilizza uno schema di etichettatura coerente per tutte le tue AWS risorse. Per le migliori pratiche, consulta il white paper sulle migliori pratiche di etichettatura.

Rivedi le nozioni di base sui tag

Puoi trovare i segreti in base ai tag nella console AWS CLI, e SDKs. AWS fornisce inoltre lo strumento Resource Groups per creare una console personalizzata che consolida e organizza le risorse in base ai relativi tag. Per individuare i segreti con un tag specifico, consulta the section called "Scopri i segreti".

Puoi utilizzare la console Secrets Manager o AWS CLI l'API Secrets Manager per:

- Crea un segreto con tag
- Aggiungi tag a un segreto
- Elenca i tag per i tuoi segreti
- Rimozione di tag da un segreto

Puoi usare i tag per classificare i tuoi segreti. Ad esempio, puoi classificare i segreti per scopo, proprietario o ambiente. Poiché definisci una chiave e un valore per ogni tag, puoi creare un set di categorie personalizzate per soddisfare esigenze specifiche. Di seguito sono riportati vari esempi di tag:

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

Tieni traccia dei costi utilizzando i tag

Puoi utilizzare i tag per classificare e tenere traccia AWS dei costi. Quando applichi tag alle tue AWS risorse, inclusi i segreti, il rapporto sull'allocazione AWS dei costi include l'utilizzo e i costi aggregati per tag. Puoi applicare i tag che rappresentano categorie di business (come centri di costo, nomi di applicazioni o proprietari) per organizzare i costi tra più servizi. Per ulteriori informazioni, consulta Utilizzo dei tag per l'allocazione dei costi ai fini dei report di fatturazione personalizzati nella AWS Billing User Guide (Guida per l'utente di Amazon API Gateway).

Comprendi le restrizioni relative ai tag

Ai tag si applicano le limitazioni seguenti.

Limitazioni di base

- Il numero massimo di tag per risorsa (segreta) è 50.
- Per le chiavi e i valori dei tag viene fatta la distinzione tra maiuscole e minuscole.
- Non puoi cambiare o modificare i tag per un segreto eliminato.

Limitazioni applicate alle chiavi di tag

- Ogni chiave di tag deve essere univoca. Se aggiungi un tag con una chiave già in uso, il nuovo tag sovrascrive la coppia chiave-valore esistente.
- Non puoi iniziare una chiave di tag con `aws:` perché questo prefisso è riservato all'uso di AWS. AWS crea tag che iniziano con questo prefisso per tuo conto, ma non puoi modificarli o eliminarli.
- Le chiavi di tag devono avere una lunghezza compresa tra 1 e 128 caratteri Unicode.
- Le chiavi di tag devono contenere i seguenti caratteri: lettere Unicode, cifre, spazio e i seguenti caratteri speciali: `_ . / = + - @`.

Limitazioni applicate ai valori dei tag

- I valori dei tag devono avere una lunghezza compresa tra 0 e 255 caratteri Unicode.
- I valori dei tag possono essere vuoti. In caso contrario, devono contenere i seguenti caratteri: lettere Unicode, cifre, spazio e i seguenti caratteri speciali: `_ . / = + - @`.

Etichetta i segreti utilizzando la console Secrets Manager

Puoi gestire i tag per i tuoi segreti utilizzando la console Secrets Manager.

Per accedere alle funzionalità di tagging, procedi come segue:

1. Apri la console Secrets Manager.
2. Nella barra di navigazione, scegli la tua regione preferita.
3. Nella pagina Segreti, seleziona un segreto.

Per visualizzare i tag di un segreto

- Nella pagina Dettagli segreti, scegli la scheda Tag.

Per creare un segreto con un tag

- Segui la procedura riportata in Crea segreti.

Per aggiungere o modificare i tag di un segreto

1. Nella pagina Dettagli segreti, scegli la scheda Tag, quindi scegli Modifica tag.
2. Inserisci la chiave del tag nel campo Chiave. Facoltativamente, inserisci un valore di tag nel campo Valore.
3. Scegli Save (Salva). Il tag nuovo o aggiornato viene visualizzato nell'elenco dei tag.

> **Note**
>
> Se il pulsante Salva non è abilitato, la chiave o il valore del tag potrebbero non soddisfare le restrizioni del tag. Per ulteriori informazioni, consulta Comprendi le restrizioni relative ai tag.

Per rimuovere un tag da un segreto

1. Nella pagina dei dettagli segreti, scegli la scheda Tag, quindi scegli l'icona Rimuovi accanto al tag che desideri rimuovere.
2. Scegli Salva per confermare la rimozione o seleziona Annulla per annullare.

Contrassegna i segreti usando il AWS CLI

AWS CLI esempi

Example Aggiunta di un tag a un segreto

L'esempio di `tag-resource` seguente mostra come collegare un tag con una sintassi abbreviata.

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

Example Aggiunta di più tag a un segreto

L'esempio di `tag-resource` seguente mostra come collegare due tag chiave-valore a un segreto.

```
aws secretsmanager tag-resource \
            --secret-id MyTestSecret \
```

```
            --tags '[{"Key": "FirstTag", "Value": "FirstValue"}, {"Key": "SecondTag",
 "Value": "SecondValue"}]'
```

Example Rimozione di tag da un segreto

L'esempio di `untag-resource` seguente mostra come rimuovere due tag da un segreto. Per ogni tag, vengono rimossi sia la chiave che il valore.

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

Etichetta i segreti utilizzando l'API Secrets Manager

Puoi aggiungere, elencare e rimuovere tag utilizzando l'API Secrets Manager. Per alcuni esempi, consultare la seguente documentazione:

- ListSecrets: Consente `ListSecrets` di visualizzare i tag applicati a un segreto
- TagResource: aggiunge tag a un segreto
- Rimuovi tag: rimuove i tag da un segreto

Etichetta i segreti utilizzando l' AWS SDK di Secrets Manager

Per modificare i tag del tuo segreto, utilizza le seguenti operazioni API:

- ListSecrets: `ListSecrets` Da utilizzare per visualizzare i tag applicati a un segreto
- `TagResource`: aggiunge tag a un segreto
- `UntagResource`: rimuove i tag da un segreto

Per ulteriori informazioni sull'utilizzo dell'SDK, consultathe section called “AWS SDKs”.

Replica i Gestione dei segreti AWS segreti in tutte le regioni

Puoi replicare i tuoi segreti in più aree Regioni AWS per supportare le applicazioni distribuite in quelle regioni e soddisfare i requisiti di accesso regionali e di bassa latenza. Se necessario in un secondo momento, è possibile promuovere un segreto di replica a uno standalone e quindi configurarlo per la replica in modo indipendente. Secrets Manager replica i dati segreti crittografati e i metadati, ad esempio tag e policy delle risorse tra le Regioni specificate.

L'ARN di un segreto replicato è lo stesso del segreto principale ad eccezione della regione, ad esempio:

- Segreto primario: `arn:aws:secretsmanager:Region1:123456789012:secret:MySecret-a1b2c3`
- Segreto di replica: `arn:aws:secretsmanager:Region2:123456789012:secret:MySecret-a1b2c3`

Per informazioni sui prezzi dei segreti di replica, consulta Prezzi di Gestione dei segreti AWS.

Quando archivi le credenziali del database per un database di origine replicato in altre regioni, il segreto contiene informazioni di connessione per il database di origine. Se poi replichi il segreto, le repliche saranno copie del segreto di origine e conterranno le stesse informazioni di connessione. È possibile aggiungere altre key/value coppie al segreto per le informazioni di connessione regionali.

Se si attiva il segreto primario per la rotazione, Secrets Manager esegue la rotazione segreta nella regione principale e il nuovo valore segreto si propaga a tutti i segreti di replica associati. Non è possibile gestire la rotazione singolarmente per tutti i segreti di replica.

Puoi replicare i segreti in tutte le AWS regioni abilitate. Tuttavia, se utilizzi Secrets Manager in AWS regioni speciali come AWS GovCloud (US) o regioni della Cina, puoi configurare i segreti e le repliche solo all'interno di queste AWS regioni specializzate. Non è possibile replicare un segreto nelle AWS regioni abilitate in una regione specializzata o replicare un segreto da una regione specializzata a una regione commerciale.

Prima di poter replicare un segreto in un'altra regione, devi abilitare tale regione. Per ulteriori informazioni, consulta Gestire AWS Regioni.

Puoi utilizzare un segreto in più regioni senza replicarlo chiamando l'endpoint di Secrets Manager nella regione in cui è archiviato il segreto. Per un elenco di endpoint, consulta the section called

"Endpoint di Secrets Manager". Per utilizzare la replica per migliorare la resilienza del carico di lavoro, consulta l'articolo Disaster Recovery (DR) Architecture on AWS, Part I: Strategies for Recovery in the Cloud.

Secrets Manager genera una voce di CloudTrail registro quando si replica un segreto. Per ulteriori informazioni, consulta the section called "Accedi con AWS CloudTrail ".

Come replicare un segreto in altre regioni (console)

1. Apri la console Secrets Manager all'indirizzo https://console.aws.amazon.com/secretsmanager/.
2. Dall'elenco dei segreti, scegli il segreto.
3. Nella pagina dei dettagli del segreto, nella sezione Replica completa una delle seguenti operazioni:
   - Se il tuo segreto non viene replicato, scegli Replica il segreto.
   - Se il tuo segreto viene replicato, nella sezione Replica il segreto, scegli Aggiungi regione.
4. Nella finestra di dialogo Aggiungi valore di registro, effettua una delle operazioni indicate di seguito:

   a. Per Regione AWS , scegli la regione in cui desideri replicare il segreto.

   b. (Opzionale) In Chiave di crittografia, scegli una chiave KMS con cui crittografare il segreto. La chiave deve trovarsi nella Regione della replica.

   c. (Facoltativo) Per aggiungere un'altra regione, scegli Aggiungi altre regioni.

   d. Scegliere Replicate (Replica).

   Torna alla pagina dei dettagli del segreto. Nella sezione Replica segreto, lo Stato di replica viene visualizzato per ogni regione.

AWS CLI

Example Replica di un segreto in un'altra Regione

Nell'esempio `replicate-secret-to-regions` seguente, un segreto viene replicato nella Regione eu-west-3. La replica è crittografata con la chiave AWS aws/secretsmanager gestita.

```
aws secretsmanager replicate-secret-to-regions \
        --secret-id MyTestSecret \
```

```
        --add-replica-regions Region=eu-west-3
```

Example Crea un segreto e replicalo

L'esempio seguente crea un segreto e lo replica in eu-west-3. La replica è crittografata con. Chiave gestita da AWS aws/secretsmanager

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

AWS SDK

Per replicare un segreto, utilizzare il comando `ReplicateSecretToRegions`. Per ulteriori informazioni, consulta the section called "AWS SDKs".

Promuovi una replica segreta a una copia segreta autonoma in Gestione dei segreti AWS

Un segreto di replica è un segreto che viene replicato da un segreto primario a un altro. Regione AWS Ha lo stesso valore segreto e metadati del primario, ma può essere crittografato con una chiave KMS diversa. Un segreto di replica non può essere aggiornato indipendentemente dal segreto principale, con l'eccezione della chiave di crittografia. La promozione di un segreto di replica disconnette tale segreto da quello primario e rende la replica segreta un segreto autonomo. Modifiche al segreto primario non vengono replicate nel segreto autonomo.

Puoi promuovere un segreto di replica a un segreto autonomo come soluzione di ripristino di emergenza in caso di errore del segreto primario. In alternativa, è utile promuovere una replica in un segreto autonomo se si desidera attivare la rotazione per la replica.

Se si promuove una replica, per utilizzare il segreto autonomo, assicurati di aggiornare le applicazioni corrispondenti.

Secrets Manager genera una voce di CloudTrail registro quando promuovi un segreto. Per ulteriori informazioni, consulta the section called "Accedi con AWS CloudTrail ".

Per promuovere un segreto di replica (console)

1. Accedere a Secrets Manager all'indirizzo https://console.aws.amazon.com/secretsmanager/.
2. Accesso alla Regione di replica.
3. Nella pagina Secrets (Segreti), scegli il segreto di replica.
4. Nella pagina dei dettagli dei segreti di replica, scegli Promote to standalone secret (Promuovi a segreto autonomo).
5. Nella finestra di dialogo Promote replica to standalone secret (Promuovi replica a segreto autonomo), inserisci la Regione e quindi scegli Promote replica (Promuovi replica).

AWS CLI

Example Promozione di un segreto di replica a primario

L'esempio di `stop-replication-to-replica` seguente mostra come rimuovere il collegamento tra un segreto di replica e quello primario. Il segreto di replica viene promosso a segreto primario nella Regione della replica. È necessario effettuare una chiamata `stop-replication-to-replica` dall'interno della Regione della replica.

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

AWS SDK

Per promuovere una replica in un segreto autonomo, usare il comando `StopReplicationToReplica`. È necessario richiamare questa azione dalla regione segreta di replica. Per ulteriori informazioni, consulta the section called "AWS SDKs".

Impedisci la Gestione dei segreti AWS replica

Poiché i segreti possono essere replicati utilizzando `ReplicateSecretToRegions`o quando vengono creati utilizzando `CreateSecret`, se desideri impedire agli utenti di replicare i segreti, ti consigliamo di impedire le azioni che contengono il parametro. `AddReplicaRegions` È possibile utilizzare un'`Condition`istruzione nelle politiche di autorizzazione per consentire solo azioni che non aggiungono aree di replica. Consulta i seguenti esempi di policy per le istruzioni Condition che puoi utilizzare.

Example Impedisci l'autorizzazione alla replica

Il seguente esempio di policy mostra come consentire tutte le azioni che non aggiungono aree di replica. Ciò impedisce agli utenti di replicare i segreti tramite entrambi `ReplicateSecretToRegions` e. `CreateSecret`

JSON

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

Example Consenti l'autorizzazione di replica solo a regioni specifiche

La seguente politica mostra come consentire tutte le seguenti operazioni:

- Crea segreti senza repliche
- Crea segreti con replica nelle regioni solo negli Stati Uniti d'America e in Canada
- Replica i segreti nelle regioni solo negli Stati Uniti d'America e in Canada

JSON

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

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

Risoluzione dei problemi relativi Gestione dei segreti AWS alla replica

Gestione dei segreti AWS la replica potrebbe fallire per vari motivi. Per verificare il motivo per cui un segreto non è stato replicato, puoi effettuare una delle seguenti operazioni:

- Chiama l'operazione `DescribeSecret` API
- Rivedi AWS CloudTrail gli eventi

Quando la replica fallisce:

- Se non ci sono versioni segrete utilizzabili, Secrets Manager rimuove il segreto dalla regione di replica.
- Se esistono versioni segrete replicate correttamente, queste rimangono nella regione di replica finché non vengono rimosse esplicitamente utilizzando l'operazione API. `RemoveRegionsFromReplication`

Le sezioni seguenti descrivono alcuni motivi comuni degli errori di replica.

Esiste un segreto con lo stesso nome nella Regione selezionata

Per risolvere questo problema è possibile sovrascrivere il nome duplicato del segreto nella Regione di replica. Riprova la replica e poi nella casella di dialogo Riprova la replica scegli Sovrascrivi.

Nessuna autorizzazione disponibile sulla chiave KMS per completare la replica

Secrets Manager innanzitutto crittografa il segreto prima di crittografarlo nuovamente con la nuova chiave KMS nella Regione di replica. Se non disponi dell'autorizzazione `kms:Decrypt` per la chiave di crittografia nella Regione principale, riscontrerai questo errore. Per crittografare il segreto replicato con una chiave KMS diversa da `aws/secretsmanager`, è necessario `kms:GenerateDataKey` e `kms:Encrypt` alla chiave. Per informazioni, consulta the section called "Autorizzazioni per la chiave KMS".

La chiave KMS è stata disattivata o non è stata trovata

Se la chiave di crittografia nella regione principale è disabilitata o eliminata, Secrets Manager non può replicare il segreto. Se il segreto presenta versioni con etichetta personalizzata crittografate con la chiave di crittografia disabilitata o eliminata, questo errore può verificarsi anche se la chiave di crittografia è stata modificata. Per informazioni su come Secrets Manager esegue la crittografia, consulta the section called "Crittografia e decrittografia del segreto". Per risolvere questo problema, è possibile ricreare le versioni segrete in modo che Secrets Manager possa crittografarle con la chiave di crittografia corrente. Per ulteriori informazioni, consulta Modificare la chiave di crittografia per un segreto. Quindi riprova la replica.

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

Non è stata abilitata la Regione in cui si verifica la replica

Per informazioni su come abilitare una Regione, consulta Gestione delle Regioni AWS nella Guida di riferimento alla gestione degli account AWS .

Ottieni segreti da Gestione dei segreti AWS

Secrets Manager genera una voce di CloudTrail registro quando si recupera un segreto. Per ulteriori informazioni, consulta the section called "Accedi con AWS CloudTrail ".

È possibile recuperare valori segreti utilizzando:

- Ottieni un valore segreto di Secrets Manager usando Java
- Ottieni un valore segreto di Secrets Manager usando Python
- Ottieni un valore segreto di Secrets Manager utilizzando .NET
- Ottieni un valore segreto di Secrets Manager usando Go
- Ottieni un valore segreto di Secrets Manager usando Rust
- Usa Gestione dei segreti AWS i segreti in Amazon Elastic Kubernetes Service
- Usa Gestione dei segreti AWS i segreti nelle AWS Lambda funzioni
- Utilizzo dell' Gestione dei segreti AWS agente
- Ottieni un valore segreto di Secrets Manager utilizzando l'SDK C++ AWS
- Ottieni un valore segreto di Secrets Manager utilizzando l' JavaScript AWS SDK
- Ottieni un valore segreto di Secrets Manager utilizzando l'SDK Kotlin AWS
- Ottieni un valore segreto di Secrets Manager utilizzando l'SDK PHP AWS
- Ottieni un valore segreto di Secrets Manager usando Ruby SDK AWS
- Ottieni un valore segreto usando il AWS CLI
- Ottieni un valore segreto usando la console AWS
- Usa Gestione dei segreti AWS i segreti in AWS Batch
- Ottieni un Gestione dei segreti AWS segreto in una CloudFormation risorsa
- Usa Gestione dei segreti AWS i segreti nei GitHub lavori
- Usa Gestione dei segreti AWS in GitLab
- Usa Gestione dei segreti AWS i segreti in AWS IoT Greengrass
- Usa Gestione dei segreti AWS i segreti in Parameter Store

Ottieni un valore segreto di Secrets Manager usando Java

Nelle applicazioni, puoi recuperare i tuoi segreti chiamando `GetSecretValue` o `BatchGetSecretValue` in uno qualsiasi dei AWS SDKs. Tuttavia, ti consigliamo di memorizzare

nella cache i valori del segreto utilizzando la caching lato client. Memorizzare i segreti nella cache migliora la velocità e riduce i costi.

Per connettersi a un database utilizzando le credenziali di un segreto, è possibile utilizzare i driver Secrets Manager SQL Connection, che racchiudono il driver JDBC di base. Questa soluzione utilizza anche la memorizzazione nella cache lato client, in modo da ridurre i costi di chiamata a Secrets Manager. APIs

Argomenti

- Ottieni un valore segreto di Secrets Manager utilizzando Java con memorizzazione nella cache lato client
- Connect a un database SQL utilizzando JDBC con credenziali segrete Gestione dei segreti AWS
- Ottieni un valore segreto di Secrets Manager utilizzando Java AWS SDK

Ottieni un valore segreto di Secrets Manager utilizzando Java con memorizzazione nella cache lato client

Quando si recupera un segreto, è possibile utilizzare il componente di caching basato su Java di Secrets Manager per memorizzarlo nella cache per un uso futuro. Il recupero di un segreto memorizzato nella cache è più veloce rispetto al recupero da Secrets Manager. Poiché la chiamata a Secrets Manager comporta un costo APIs, l'utilizzo di una cache può ridurre i costi. Per tutti i modi in cui puoi recuperare i segreti, vedi Ottieni segreti.

La policy della cache è Least Recently Used (LRU), quindi quando la cache deve eliminare un segreto, elimina il segreto usato meno di recente. Di default, la cache aggiorna i segreti ogni ora. È possibile configurare la frequenza con cui il segreto viene aggiornato nella cache ed è possibile collegarsi al recupero del segreto per aggiungere altre funzionalità.

La cache non impone la rimozione di oggetti inutili (garbage collection) una volta liberati i riferimenti alla cache. L'implementazione della cache non include l'invalidazione della cache. L'implementazione della cache è incentrata sulla cache stessa e non è rafforzata o focalizzata sulla sicurezza. Se hai bisogno di un livello di sicurezza aggiuntivo, come la crittografia degli elementi nella cache, usa le interfacce e i metodi astratti forniti.

Per usare il componente, devi disporre dei seguenti elementi:

- Ambiente di sviluppo Java 8 o versioni successive. Consulta Java SE Downloads sul sito Web di Oracle.

Per scaricare il codice sorgente, vedete Secrets Manager, componente client di caching basato su Java su. GitHub

Per aggiungere il componente al progetto, nel file Maven pom.xml, includi la seguente dipendenza. Per ulteriori informazioni su Maven, consulta la Guida alle operazioni di base sul sito Web Apache Maven Project.

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

Autorizzazioni richieste:

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

Per ulteriori informazioni, consulta Riferimento per le autorizzazioni.

Documentazione di riferimento

- SecretCache
- SecretCacheConfiguration
- SecretCacheHook

Example Recupero di un segreto

L'esempio di codice riportato di seguito mostra una funzione Lambda che recupera una stringa del segreto. Segue la best practice di creare un'istanza della cache al di fuori del gestore della funzione quindi non continua a chiamare l'API se si chiama nuovamente la funzione Lambda.

```
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

Una cache in memoria per i segreti richiesti da Secrets Manager. Si usa the section called "getSecretString" o the section called "getSecretBinary" per recuperare un segreto dalla cache. È possibile configurare le impostazioni della cache specificando un oggetto the section called "SecretCacheConfiguration" nel costruttore.

Per ulteriori informazioni, inclusi esempi, consulta the section called "Java con memorizzazione nella cache lato client".

Costruttori

`public SecretCache()`

Costruttore di default per un oggetto `SecretCache`.

`public SecretCache(AWSSecretsManagerClientBuilder builder)`

Costruisce una nuova cache utilizzando un client di Secrets Manager creato utilizzando l'`AWSSecretsManagerClientBuilder` fornito. Utilizzate questo costruttore per personalizzare il client Secrets Manager, ad esempio per utilizzare una regione o un endpoint specifici.

`public SecretCache(AWSSecretsManager client)`

Costruisce una nuova cache del segreto utilizzando l'`AWSSecretsManagerClient` fornito. Utilizzate questo costruttore per personalizzare il client Secrets Manager, ad esempio per utilizzare una regione o un endpoint specifici.

`public SecretCache(SecretCacheConfiguration config)`

Costruisce una nuova cache del segreto utilizzando il `the section called "SecretCacheConfiguration"` fornito.

Metodi

getSecretString

```
public String getSecretString(final String secretId)
```

Recupera un segreto stringa da Secrets Manager. Restituisce una `String`.

getSecretBinary

```
public ByteBuffer getSecretBinary(final String secretId)
```

Recupera un segreto binario da Secrets Manager. Restituisce un `ByteBuffer`.

refreshNow

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

Forza l'aggiornamento della cache. Restituisce `true` se l'aggiornamento è stato completato senza errori, altrimenti `false`.

close

```
public void close()
```

Chiude la cache.

SecretCacheConfiguration

Opzioni di configurazione della cache per una the section called "SecretCache", ad esempio dimensione massima della cache e durata (TTL) per i segreti memorizzati nella cache.

Costruttore

```
public SecretCacheConfiguration
```

Costruttore di default per un oggetto `SecretCacheConfiguration`.

Metodi

getClient

```
public AWSSecretsManager getClient()
```

Restituisce l'`AWSSecretsManagerClient` da cui la cache recupera segreti.

setClient

```
public void setClient(AWSSecretsManager client)
```

Restituisce il client `AWSSecretsManagerClient` da cui la cache recupera segreti.

getCacheHook

```
public SecretCacheHook getCacheHook()
```

Restituisce l'interfaccia the section called "SecretCacheHook" utilizzata per collegarsi agli aggiornamenti della cache.

setCacheHook

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

Imposta l'interfaccia the section called "SecretCacheHook" utilizzata per collegarsi agli aggiornamenti della cache.

getMaxCacheDimensioni

```
public int getMaxCacheSize()
```

Restituisce la dimensione massima della cache. Il valore di default è 1024 segreti.

setMaxCacheDimensioni

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

Imposta la dimensione massima della cache. Il valore di default è 1024 segreti.

getCacheItemTTL

```
public long getCacheItemTTL()
```

Restituisce il TTL in millisecondi per gli elementi memorizzati nella cache. Quando un segreto memorizzato nella cache supera questo TTL, la cache recupera una nuova copia del segreto dal `AWSSecretsManagerClient`. Il valore predefinito è 1 ora in millisecondi.

La cache aggiorna il segreto in modo sincrono quando viene richiesto il segreto dopo il TTL. Se l'aggiornamento sincrono ha esito negativo, la cache restituisce il segreto non aggiornato.

setCacheItemTTL

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

Restituisce il TTL in millisecondi per gli elementi memorizzati nella cache. Quando un segreto memorizzato nella cache supera questo TTL, la cache recupera una nuova copia del segreto dal `AWSSecretsManagerClient`. Il valore predefinito è 1 ora in millisecondi.

getVersionStage

```
public String getVersionStage()
```

Restituisce la versione dei segreti che si desidera memorizzare nella cache. Per ulteriori informazioni, consulta Versioni del segreto. Il valore predefinito è `"AWSCURRENT"`.

setVersionStage

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

Imposta la versione dei segreti che si desidera memorizzare nella cache. Per ulteriori informazioni, consulta Versioni del segreto. Il valore predefinito è `"AWSCURRENT"`.

SecretCacheConfiguration Con Client

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

Imposta il `AWSSecretsManagerClient` da cui recuperare segreti. Restituisce l'oggetto `SecretCacheConfiguration` aggiornato con la nuova impostazione.

SecretCacheConfiguration withCacheHook

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

Imposta l'interfaccia utilizzata per collegare la cache in memoria. Restituisce l'oggetto `SecretCacheConfiguration` aggiornato con la nuova impostazione.

SecretCacheConfiguration withMaxCacheDimensioni

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

Imposta la dimensione massima della cache. Restituisce l'oggetto `SecretCacheConfiguration` aggiornato con la nuova impostazione.

SecretCacheConfiguration withCacheItemTTL

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

Restituisce il TTL in millisecondi per gli elementi memorizzati nella cache. Quando un segreto memorizzato nella cache supera questo TTL, la cache recupera una nuova copia del segreto dal `AWSSecretsManagerClient`. Il valore predefinito è 1 ora in millisecondi. Restituisce l'oggetto `SecretCacheConfiguration` aggiornato con la nuova impostazione.

SecretCacheConfiguration withVersionStage

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

Imposta la versione dei segreti che si desidera memorizzare nella cache. Per ulteriori informazioni, consulta Versioni del segreto. Restituisce l'oggetto `SecretCacheConfiguration` aggiornato con la nuova impostazione.

SecretCacheHook

Un'interfaccia per collegarsi a una the section called "SecretCache" per eseguire operazioni sui segreti memorizzati al suo interno.

put

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

Prepara l'oggetto per la memorizzazione nella cache.

Restituisce l'oggetto da memorizzare nella cache.

get

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

Deriva l'oggetto dall'oggetto memorizzato nella cache.

Restituisce l'oggetto da restituire dalla cache

Connect a un database SQL utilizzando JDBC con credenziali segrete Gestione dei segreti AWS

Nelle applicazioni Java, è possibile utilizzare i driver Secrets Manager SQL Connection per connettersi ai database MySQL, PostgreSQL, MSSQLServer Oracle, Db2 e Redshift utilizzando

le credenziali archiviate in Secrets Manager. Ogni driver esegue il wrapping del driver JDBC di base per consentire l'utilizzo delle chiamate JDBC per accedere al database. Tuttavia, invece di specificare un nome utente e una password per la connessione, si fornisce l'ID di un segreto. Il driver chiama Secrets Manager per recuperare il valore del segreto, quindi utilizza le credenziali nel segreto per connettersi al database. Il driver inoltre memorizza le credenziali nella cache utilizzando la libreria di caching lato client Java, in modo che per le connessioni future non sia necessaria una chiamata a Secrets Manager. Per impostazione predefinita, la cache viene aggiornata ogni ora e anche quando un segreto viene ruotato. Per configurare la cache, consulta the section called "SecretCacheConfiguration".

È possibile scaricare il codice sorgente da. GitHub

Per utilizzare i driver di connessione SQL di Secrets Manager:

- L'applicazione deve essere in Java 8 o versioni successive.
- Il segreto deve essere uno fra i seguenti:
  - Un segreto di database nella struttura JSON prevista. Per verificare il formato, nella console di Secrets Manager, visualizza il tuo segreto e seleziona Retrieve secret value (Recupera valore segreto). In alternativa AWS CLI, nella chiamata get-secret-value.
  - Un segreto gestito da Amazon RDS. Per questo tipo di segreto, quando si stabilisce la connessione è necessario specificare un endpoint e una porta.
  - Un segreto gestito da Amazon Redshift. Per questo tipo di segreto, quando si stabilisce la connessione è necessario specificare un endpoint e una porta.

Se il database viene replicato in altre regioni, per connettersi a un database di replica in una regione differente, è necessario specificare l'endpoint e la porta della regione al momento della creazione della connessione. Puoi memorizzare le informazioni di connessione regionali nel segreto come key/value coppie aggiuntive, nei parametri SSM Parameter Store o nella configurazione del codice.

Per aggiungere il driver al progetto, nel file di build Maven `pom.xml`, aggiungi la seguente dipendenza per il driver. Per ulteriori informazioni, consulta Secrets Manager SQL Connection Library sul sito Web di Maven Central Repository.

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

Il driver utilizza la catena di provider delle credenziali predefinita. Se esegui il driver su Amazon EKS, potrebbe raccogliere le credenziali del nodo su cui è in esecuzione anziché il ruolo dell'account di servizio. Per risolvere questo problema, aggiungi la versione 1 di `com.amazonaws:aws-java-sdk-sts` al tuo file di progetto Gradle o Maven come dipendenza.

Per impostare un URL di endpoint AWS PrivateLink DNS e una regione nel file: `secretsmanager.properties`

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

Per sovrascrivere la regione primaria, imposta la variabile d'ambiente `AWS_SECRET_JDBC_REGION` o apporta la seguente modifica al file `secretsmanager.properties`:

```
drivers.region = region
```

Autorizzazioni richieste:

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

Per ulteriori informazioni, consulta Riferimento per le autorizzazioni.

Esempi:

- Stabilire una connessione a un database
- Impostazione di una connessione specificando l'endpoint e la porta
- Utilizzare il pooling di connessioni c3p0 per stabilire una connessione
- Utilizzo del pooling di connessioni c3p0 per stabilire una connessione specificando l'endpoint e la porta

Stabilire una connessione a un database

Nell'esempio seguente viene illustrato come stabilire una connessione a un database utilizzando le credenziali e le informazioni di connessione in un segreto. Una volta acquisita la connessione, puoi utilizzare le chiamate JDBC per accedere al database. Per ulteriori informazioni, consulta JDBC Basics sul sito Web della documentazione Java.

MySQL

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

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

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

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

PostgreSQL

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

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

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

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

Oracle

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

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

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

```
info.put( "user", "secretId" );

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

MSSQLServer

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

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

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

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

Db2

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

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

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

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

Redshift

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

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

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

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

Impostazione di una connessione specificando l'endpoint e la porta

Nell'esempio seguente viene illustrato come stabilire una connessione a un database utilizzando le credenziali in un segreto con un endpoint e una porta specificati dall'utente.

I segreti gestiti da Amazon RDS non includono l'endpoint e la porta del database. Per connettersi a un database utilizzando le credenziali master in un segreto gestito da Amazon RDS, è necessario specificarle nel codice.

I segreti replicati in altre Regioni possono migliorare la latenza per la connessione al database regionale, ma non contengono informazioni di connessione diverse dal segreto di origine. Ogni replica è una copia del segreto di origine. Per memorizzare le informazioni di connessione regionali in modalità segreta, aggiungi altre key/value coppie per l'endpoint e le informazioni sulla porta per le regioni.

Una volta acquisita la connessione, puoi utilizzare le chiamate JDBC per accedere al database. Per ulteriori informazioni, consulta JDBC Basics sul sito Web della documentazione Java.

MySQL

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

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

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

```
info.put( "user", "secretId" );

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

PostgreSQL

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

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

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

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

Oracle

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

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

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

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

MSSQLServer

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

```
Class.forName( "com.amazonaws.secretsmanager.sql.AWSSecretsManagerMSSQLServerDriver" ).newIn

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

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

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

Db2

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

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

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

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

Redshift

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

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

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

```
info.put( "user", "secretId" );

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

Utilizzare il pooling di connessioni c3p0 per stabilire una connessione

Nell'esempio seguente viene illustrato come stabilire un pool di connessioni con un file `c3p0.properties` che utilizza il driver per recuperare le credenziali e le informazioni sulla connessione dal segreto. Per `user` e `jdbcUrl`, inserisci l'ID segreto per configurare il pool di connessioni. Puoi quindi recuperare le connessioni dal pool e utilizzarle come qualsiasi altra connessione al database. Per ulteriori informazioni, consulta JDBC Basics sul sito Web della documentazione Java.

Per ulteriori informazioni su c3p0, consulta c3p0 sul sito 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
```

Utilizzo del pooling di connessioni c3p0 per stabilire una connessione specificando l'endpoint e la porta

L'esempio seguente mostra come stabilire un pool di connessioni con un `c3p0.properties` file che utilizza il driver per recuperare le credenziali in un segreto con un endpoint e una porta specificati dall'utente. Puoi quindi recuperare le connessioni dal pool e utilizzarle come qualsiasi altra connessione al database. Per ulteriori informazioni, consulta JDBC Basics sul sito Web della documentazione Java.

I segreti gestiti da Amazon RDS non includono l'endpoint e la porta del database. Per connettersi a un database utilizzando le credenziali master in un segreto gestito da Amazon RDS, è necessario specificarle nel codice.

I segreti replicati in altre Regioni possono migliorare la latenza per la connessione al database regionale, ma non contengono informazioni di connessione diverse dal segreto di origine. Ogni replica è una copia del segreto di origine. Per memorizzare le informazioni di connessione regionali nel segreto, aggiungi altre key/value coppie per l'endpoint e le informazioni sulla porta per le regioni.

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
```

Ottieni un valore segreto di Secrets Manager utilizzando Java AWS SDK

Nelle applicazioni, puoi recuperare i tuoi segreti chiamando `GetSecretValue` o `BatchGetSecretValue` in uno qualsiasi dei. AWS SDKs Tuttavia, ti consigliamo di memorizzare nella cache i valori del segreto utilizzando la caching lato client. Memorizzare i segreti nella cache migliora la velocità e riduce i costi.

- Se memorizzi le credenziali del database nel segreto, utilizza i driver di connessione SQL di Secrets Manager per eseguire la connessione a un database utilizzando le credenziali nel segreto.
- Per altri tipi di segreti, usa il componente di caching basato su Java Secrets Manager o chiama l'SDK direttamente con o. `GetSecretValue``BatchGetSecretValue`

Gli esempi di codice seguenti mostrano come utilizzare `GetSecretValue`.

Autorizzazioni richieste:`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);
        }
    }
}
```

Ottieni un valore segreto di Secrets Manager usando Python

Nelle applicazioni, puoi recuperare i tuoi segreti chiamando `GetSecretValue` o `BatchGetSecretValue` in uno qualsiasi dei. AWS SDKs Tuttavia, ti consigliamo di memorizzare nella cache i valori del segreto utilizzando la caching lato client. Memorizzare i segreti nella cache migliora la velocità e riduce i costi.

Argomenti

- Ottieni un valore segreto di Secrets Manager usando Python con caching lato client
- Ottieni un valore segreto di Secrets Manager usando l'SDK Python AWS
- Ottieni un batch di valori segreti di Secrets Manager usando Python SDK AWS

Ottieni un valore segreto di Secrets Manager usando Python con caching lato client

Quando si recupera un segreto, è possibile utilizzare il componente di caching basato su Python di Secrets Manager per memorizzarlo nella cache per un uso futuro. Il recupero di un segreto memorizzato nella cache è più veloce rispetto al recupero da Secrets Manager. Poiché la chiamata a Secrets Manager comporta un costo APIs, l'utilizzo di una cache può ridurre i costi. Per tutti i modi in cui puoi recuperare i segreti, vedi Ottieni segreti.

La policy della cache è Least Recently Used (LRU), quindi quando la cache deve eliminare un segreto, elimina il segreto usato meno di recente. Di default, la cache aggiorna i segreti ogni ora. È possibile configurare la frequenza con cui il segreto viene aggiornato nella cache ed è possibile collegarsi al recupero del segreto per aggiungere altre funzionalità.

La cache non impone la rimozione di oggetti inutili (garbage collection) una volta liberati i riferimenti alla cache. L'implementazione della cache non include l'invalidazione della cache. L'implementazione della cache è incentrata sulla cache stessa e non è rafforzata o focalizzata sulla sicurezza. Se hai bisogno di un livello di sicurezza aggiuntivo, come la crittografia degli elementi nella cache, usa le interfacce e i metodi astratti forniti.

Per usare il componente, devi disporre dei seguenti elementi:

- Python 3.6 o versioni successive.
- botocore 1.12 o versioni successive. Consulta AWS SDK for Python e Botocore.
- setuptools_scm 3.2 o versioni successive. Vedi https://pypi. org/project/setuptools-scm/.

Per scaricare il codice sorgente, consulta il componente client di caching basato su Python di Secrets Manager su. GitHub

Per installare il componente, utilizza il comando seguente.

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

Autorizzazioni richieste:

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

Per ulteriori informazioni, consulta Riferimento per le autorizzazioni.

Documentazione di riferimento

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

Example Recupero di un segreto

L'esempio seguente mostra come ottenere il valore segreto per un segreto denominato. *mysecret*

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

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

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

SecretCache

Una cache in memoria per i segreti recuperati da Secrets Manager. Si usa the section called "get_secret_string" o the section called "get_secret_binary" per recuperare un segreto dalla cache. È possibile configurare le impostazioni della cache specificando un oggetto the section called "SecretCacheConfig" nel costruttore.

Per ulteriori informazioni, inclusi esempi, consulta the section called "Python con memorizzazione nella cache lato client".

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

Questi sono i metodi disponibili:

- get_secret_string
- get_secret_binary

get_secret_string

Recupera il valore della stringa del segreto.

Sintassi della richiesta

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

Parameters

- `secret_id`(string): [Obbligatorio] Il nome o l'ARN del segreto.
- `version_stage`(stringa): La versione dei segreti che desideri recuperare. Per ulteriori informazioni, consulta Versioni segrete. Il valore di default è 'AWSCURRENT'.

Tipo restituito

stringa

get_secret_binary

Recupera il valore binario del segreto.

Sintassi della richiesta

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

Parameters

- `secret_id`(string): [Obbligatorio] Il nome o l'ARN del segreto.
- `version_stage`(stringa): La versione dei segreti che desideri recuperare. Per ulteriori informazioni, consulta Versioni segrete. Il valore di default è 'AWSCURRENT'.

Tipo restituito

Stringa con codifica Base64

SecretCacheConfig

Opzioni di configurazione della cache per una the section called "SecretCache", ad esempio dimensione massima della cache e durata (TTL) per i segreti memorizzati nella cache.

Parameters

`max_cache_size` (int)

La dimensione massima della cache. Il valore predefinito è `1024` segreti.

`exception_retry_delay_base` (int)

Il numero di secondi di attesa dopo aver riscontrato un'eccezione prima di riprovare la richiesta. Il valore predefinito è `1`.

`exception_retry_growth_factor` (int)pur

Il fattore di crescita da utilizzare per calcolare il tempo di attesa tra i tentativi di richieste non riuscite. Il valore predefinito è 2.

`exception_retry_delay_max` (int)

Il numero massimo di secondi di attesa tra le richieste non riuscite. Il valore predefinito è `3600`.

`default_version_stage` (str)

Imposta la versione dei segreti che si desidera memorizzare nella cache. Per ulteriori informazioni, consulta Versioni del segreto. Il valore predefinito è `'AWSCURRENT'`.

`secret_refresh_interval` (int)

Il numero di secondi da attendere tra gli aggiornamenti delle informazioni del segreto memorizzate nella cache. Il valore predefinito è `3600`.

`secret_cache_hook` (SecretCacheHook)

Un'implementazione della classe astratta di `SecretCacheHook`. Il valore predefinito è `None`.

SecretCacheHook

Un'interfaccia per collegarsi a una the section called "SecretCache" per eseguire operazioni sui segreti memorizzati al suo interno.

Questi sono i metodi disponibili:

- put
- get

put

Prepara l'oggetto per la memorizzazione nella cache.

Sintassi della richiesta

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

Parameters

- `obj` (object) -- [Obbligatorio] Il segreto o l'oggetto che contiene il segreto.

Tipo restituito

oggetto

get

Deriva l'oggetto dall'oggetto memorizzato nella cache.

Sintassi della richiesta

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

Parameters

- `obj`(oggetto): [Obbligatorio] Il segreto o l'oggetto che contiene il segreto.

Tipo restituito

oggetto

@InjectSecretString

Questo decoratore prevede una stringa identificativa del segreto e the section called "SecretCache" come primo e secondo argomento. Il decoratore restituisce il valore della stringa del segreto. Il segreto deve contenere una stringa.

```
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

Questo decoratore prevede una stringa identificativa del segreto e the section called "SecretCache" come primo e secondo argomento. Gli argomenti rimanenti mappano i parametri dalla funzione wrapping alle chiavi JSON nel segreto. Il segreto deve contenere una stringa nella struttura JSON.

Per un segreto che contiene questo JSON:

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

Gli esempi seguenti mostrano come estrarre i valori JSON per `username` e `password` dal segreto.

```
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')
```

Ottieni un valore segreto di Secrets Manager usando l'SDK Python AWS

Nelle applicazioni, puoi recuperare i tuoi segreti chiamando `GetSecretValue` o `BatchGetSecretValue` in uno qualsiasi dei. AWS SDKs Tuttavia, ti consigliamo di memorizzare nella cache i valori del segreto utilizzando la caching lato client. Memorizzare i segreti nella cache migliora la velocità e riduce i costi.

Per le applicazioni Python, utilizza il componente di caching basato su Python di Secrets Manager o chiama direttamente l'SDK con `get_secret_value` o `batch_get_secret_value`.

Gli esempi di codice seguenti mostrano come utilizzare `GetSecretValue`.

Autorizzazioni richieste:`secretsmanager:GetSecretValue`

```
"""
Purpose

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

from get_secret_value import GetSecretWrapper

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


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

    :param secret_name: Name of the secret to retrieve.
    :type secret_name: str
    """
    try:
```

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

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


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

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

Ottieni un batch di valori segreti di Secrets Manager usando Python SDK AWS

Il seguente esempio di codice mostra come ottenere un batch di valori segreti di Secrets Manager.

Autorizzazioni richieste:

- `secretsmanager:BatchGetSecretValue`
- `secretsmanager:GetSecretValue`autorizzazione per ogni segreto che desideri recuperare.
- Se usi i filtri, devi avere anche `secretsmanager:ListSecrets`.

Per un esempio di policy delle autorizzazioni, consulta the section called "Esempio: autorizzazione a recuperare un gruppo di valori segreti in un batch".

> **Important**
>
> Se hai una policy VPCE che nega l'autorizzazione per recuperare un singolo segreto nel gruppo su cui stai agendo, `BatchGetSecretValue` non restituirà alcun valore segreto e restituirà un errore.

```
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
```

Ottieni un valore segreto di Secrets Manager utilizzando .NET

Nelle applicazioni, puoi recuperare i tuoi segreti chiamando `GetSecretValue` o `BatchGetSecretValue` in uno qualsiasi dei AWS SDKs. Tuttavia, ti consigliamo di memorizzare nella cache i valori del segreto utilizzando la caching lato client. Memorizzare i segreti nella cache migliora la velocità e riduce i costi.

Argomenti

- Ottieni un valore segreto di Secrets Manager usando.NET con caching lato client
- Ottieni un valore segreto di Secrets Manager utilizzando SDK per .NET

Ottieni un valore segreto di Secrets Manager usando.NET con caching lato client

Quando si recupera un segreto, è possibile utilizzare il componente di caching basato su .NET di Secrets Manager per memorizzarlo nella cache per un uso futuro. Il recupero di un segreto memorizzato nella cache è più veloce rispetto al recupero da Secrets Manager. Poiché la chiamata a Secrets Manager comporta un costo APIs, l'utilizzo di una cache può ridurre i costi. Per tutti i modi in cui puoi recuperare i segreti, vedi Ottieni segreti.

La policy della cache è Least Recently Used (LRU), quindi quando la cache deve eliminare un segreto, elimina il segreto usato meno di recente. Di default, la cache aggiorna i segreti ogni ora. È possibile configurare la frequenza con cui il segreto viene aggiornato nella cache ed è possibile collegarsi al recupero del segreto per aggiungere altre funzionalità.

La cache non impone la rimozione di oggetti inutili (garbage collection) una volta liberati i riferimenti alla cache. L'implementazione della cache non include l'invalidazione della cache. L'implementazione della cache è incentrata sulla cache stessa e non è rafforzata o focalizzata sulla sicurezza. Se hai bisogno di un livello di sicurezza aggiuntivo, come la crittografia degli elementi nella cache, usa le interfacce e i metodi astratti forniti.

Per usare il componente, devi disporre dei seguenti elementi:

- .NET Framework 4.6.2 o versioni successive o .NET Standard 2.0 o versioni successive. Consulta Download .NET sul sito Web di Microsoft .NET
- L' AWS SDK per.NET. Per informazioni, consulta the section called "AWS SDKs".

Per scaricare il codice sorgente, vedi Caching client for .NET on. GitHub

Per utilizzare la cache, creane prima un'istanza, quindi recupera il segreto usando `GetSecretString` o `GetSecretBinary`. Nei recuperi successivi, la cache restituisce la copia del segreto memorizzata nella cache.

Recupero del pacchetto di caching

- Esegui una delle seguenti operazioni:
  - Nella directory del progetto, esegui il seguente comando della CLI .NET.

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

- Aggiungi il seguente riferimento al pacchetto al tuo file `.csproj`.

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

Autorizzazioni richieste:

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

Per ulteriori informazioni, consulta Riferimento per le autorizzazioni.

Documentazione di riferimento

- SecretsManagerCache
- SecretCacheConfiguration
- ISecretCacheHook

Example Recupero di un segreto

Il seguente esempio di codice mostra un metodo che recupera un segreto denominato. *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 Configurazione della durata dell'aggiornamento della cache Time To Live (TTL)

Il seguente esempio di codice mostra un metodo che recupera un segreto denominato *MySecret* e imposta la durata dell'aggiornamento della cache TTL su 24 ore.

```
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

Una cache in memoria per i segreti richiesti da Secrets Manager. Si usa the section called "GetSecretString" o the section called "GetSecretBinary" per recuperare un segreto dalla cache. È possibile configurare le impostazioni della cache specificando un oggetto the section called "SecretCacheConfiguration" nel costruttore.

Per ulteriori informazioni, inclusi esempi, consulta the section called ".NET con memorizzazione nella cache lato client".

Costruttori

`public SecretsManagerCache()`

Costruttore di default per un oggetto `SecretsManagerCache`.

`public SecretsManagerCache(IAmazonSecretsManager secretsManager)`

Costruisce una nuova cache utilizzando un client di Secrets Manager creato utilizzando l'AmazonSecretsManagerClient fornito. Utilizza questo costruttore per personalizzare il client di Secrets Manager, ad esempio per utilizzare una regione o un endpoint specifico.

Parameters

secretsManager

Il AmazonSecretsManagerClientda cui recuperare i segreti.

`public SecretsManagerCache(SecretCacheConfiguration config)`

Costruisce una nuova cache del segreto utilizzando il the section called "SecretCacheConfiguration" fornito. Utilizza questo costruttore per configurare la cache, ad esempio il numero di segreti da inserire nella cache e la frequenza di aggiornamento.

Parameters

config

Un the section called "SecretCacheConfiguration" che contiene informazioni di configurazione per la cache.

`public SecretsManagerCache(IAmazonSecretsManager secretsManager, SecretCacheConfiguration config)`

Costruisce una nuova cache utilizzando un client Secrets Manager creato utilizzando il file fornito AmazonSecretsManagerCliente unthe section called "SecretCacheConfiguration". Utilizza questo costruttore per personalizzare il client di Secrets Manager, ad esempio per utilizzare una regione o un endpoint specifici e configurare la cache, ad esempio il numero di segreti da inserire nella cache e la frequenza di aggiornamento.

Parameters

secretsManager

Il AmazonSecretsManagerClientda cui recuperare i segreti.

config

Un the section called "SecretCacheConfiguration" che contiene informazioni di configurazione per la cache.

Metodi

GetSecretString

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

Recupera un segreto stringa da Secrets Manager.

Parameters

secretId

L'ARN o il nome del segreto da recuperare.

GetSecretBinary

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

Recupera un segreto binario da Secrets Manager.

Parameters

secretId

L'ARN o il nome del segreto da recuperare.

RefreshNowAsync

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

Richiede il valore del segreto da Secrets Manager e aggiorna la cache con eventuali modifiche. Se non esiste una voce di cache esistente, ne crea una nuova. Restituisce `true` se l'aggiornamento ha esito positivo.

Parameters

secretId

L'ARN o il nome del segreto da recuperare.

GetCachedSecret

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

Restituisce la voce di cache per il segreto specificato se presente nella cache. In caso contrario, recupera il segreto da Secrets Manager e crea una nuova voce di cache.

Parameters

secretId

L'ARN o il nome del segreto da recuperare.

SecretCacheConfiguration

Opzioni di configurazione della cache per un the section called "SecretsManagerCache", come dimensione massima della cache e durata (TTL) per i segreti memorizzati nella cache.

Properties

CacheItemTTL

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

Il TTL di un elemento della cache in millisecondi. Il valore predefinito è `3600000` ms o 1 ora. Il massimo è `4294967295` ms, vale a dire circa 49,7 giorni.

MaxCacheSize

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

La dimensione massima della cache. Il valore di default è 1024 segreti. Il numero massimo è pari a 65.535.

VersionStage

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

Imposta la versione dei segreti che si desidera memorizzare nella cache. Per ulteriori informazioni, consulta Versioni del segreto. Il valore predefinito è `"AWSCURRENT"`.

Client

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

AmazonSecretsManagerClientDa cui recuperare i segreti. Se è `null`, la cache crea un'istanza di un nuovo client. Il valore predefinito è `null`.

CacheHook

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

Un the section called "ISecretCacheHook".

ISecretCacheHook

Un'interfaccia per collegarsi a una the section called "SecretsManagerCache" per eseguire operazioni sui segreti memorizzati al suo interno.

Metodi

Put

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

Prepara l'oggetto per la memorizzazione nella cache.

Restituisce l'oggetto da memorizzare nella cache.

Get

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

Deriva l'oggetto dall'oggetto memorizzato nella cache.

Restituisce l'oggetto da restituire dalla cache

Ottieni un valore segreto di Secrets Manager utilizzando SDK per .NET

Nelle applicazioni, puoi recuperare i tuoi segreti chiamando `GetSecretValue` o `BatchGetSecretValue` in uno qualsiasi dei AWS SDKs. Tuttavia, ti consigliamo di memorizzare nella cache i valori del segreto utilizzando la caching lato client. Memorizzare i segreti nella cache migliora la velocità e riduce i costi.

Per le applicazioni .NET, utilizza il componente di caching basato su .NET di Secrets Manager o chiama direttamente l'SDK con `GetSecretValue` o `BatchGetSecretValue`.

Gli esempi di codice seguenti mostrano come utilizzare `GetSecretValue`.

Autorizzazioni richieste:`secretsmanager:GetSecretValue`

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

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

        IAmazonSecretsManager client = new AmazonSecretsManagerClient();

        var response = await GetSecretAsync(client, secretName);

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

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

```
                }
            }
        }

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

            GetSecretValueResponse response = null;

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

            return response;
        }

        /// <summary>
        /// Decodes the secret returned by the call to GetSecretValueAsync and
        /// returns it to the calling program.
        /// </summary>
```

```
        /// <param name="response">A GetSecretValueResponse object containing
        /// the requested secret value returned by GetSecretValueAsync.</param>
        /// <returns>A string representing the decoded secret value.</returns>
        public static string DecodeString(GetSecretValueResponse response)
        {
            // Decrypts secret using the associated AWS Key Management Service
            // Customer Master Key (CMK.) Depending on whether the secret is a
            // string or binary value, one of these fields will be populated.
            if (response.SecretString is not null)
            {
                var secret = response.SecretString;
                return secret;
            }
            else if (response.SecretBinary is not null)
            {
                var memoryStream = response.SecretBinary;
                StreamReader reader = new StreamReader(memoryStream);
                string decodedBinarySecret =
System.Text.Encoding.UTF8.GetString(Convert.FromBase64String(reader.ReadToEnd()));
                return decodedBinarySecret;
            }
            else
            {
                return string.Empty;
            }
        }
    }
```

Ottieni un valore segreto di Secrets Manager usando Go

Nelle applicazioni, puoi recuperare i tuoi segreti chiamando `GetSecretValue` o `BatchGetSecretValue` in uno qualsiasi dei AWS SDKs. Tuttavia, ti consigliamo di memorizzare nella cache i valori del segreto utilizzando la caching lato client. Memorizzare i segreti nella cache migliora la velocità e riduce i costi.

Argomenti

- Ottieni un valore segreto di Secrets Manager usando Go con caching lato client
- Ottieni un valore segreto di Secrets Manager utilizzando Go AWS SDK

Ottieni un valore segreto di Secrets Manager usando Go con caching lato client

Quando si recupera un segreto, è possibile utilizzare il componente di caching basato su Go di Secrets Manager per memorizzarlo nella cache per un uso futuro. Il recupero di un segreto memorizzato nella cache è più veloce rispetto al recupero da Secrets Manager. Poiché la chiamata a Secrets Manager comporta un costo APIs, l'utilizzo di una cache può ridurre i costi. Per tutti i modi in cui puoi recuperare i segreti, vedi Ottieni segreti.

La policy della cache è Least Recently Used (LRU), quindi quando la cache deve eliminare un segreto, elimina il segreto usato meno di recente. Di default, la cache aggiorna i segreti ogni ora. È possibile configurare la frequenza con cui il segreto viene aggiornato nella cache ed è possibile collegarsi al recupero del segreto per aggiungere altre funzionalità.

La cache non impone la rimozione di oggetti inutili (garbage collection) una volta liberati i riferimenti alla cache. L'implementazione della cache non include l'invalidazione della cache. L'implementazione della cache è incentrata sulla cache stessa e non è rafforzata o focalizzata sulla sicurezza. Se hai bisogno di un livello di sicurezza aggiuntivo, come la crittografia degli elementi nella cache, usa le interfacce e i metodi astratti forniti.

Per usare il componente, devi disporre dei seguenti elementi:

- AWS SDK for Go. Per informazioni, consulta the section called "AWS SDKs".

Per scaricare il codice sorgente, consulta Secrets Manager Go caching client on GitHub.

Per configurare un ambiente di sviluppo Go, consulta Introduzione a Golang sul sito Web del linguaggio di programmazione Go.

Autorizzazioni richieste:

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

Per ulteriori informazioni, consulta Riferimento per le autorizzazioni.

Documentazione di riferimento

- type Cache

- tipo CacheConfig
- tipo CacheHook

Example Recupero di un segreto

L'esempio di codice riportato di seguito mostra una funzione Lambda che recupera un segreto.

```
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)
}
```

type Cache

Una cache in memoria per i segreti richiesti da Secrets Manager. Si usa the section called "GetSecretString" o the section called "GetSecretBinary" per recuperare un segreto dalla cache.

Nell'esempio seguente viene illustrato come configurare le impostazioni della 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 },
)
```

Per ulteriori informazioni, inclusi esempi, consulta the section called "Scegli la memorizzazione nella cache lato client".

Metodi

Novità

`func New(optFns ...func(*Cache)) (*Cache, error)`

New costruisce una cache del segreto utilizzando opzioni funzionali, altrimenti usa i valori predefiniti. Inizializza un SecretsManager client da una nuova sessione. Inizializza CacheConfig ai valori predefiniti. Inizializza la cache LRU con una dimensione massima predefinita.

GetSecretString

`func (c *Cache) GetSecretString(secretId string) (string, error)`

GetSecretString ottiene il valore della stringa segreta dalla cache per un determinato ID segreto. Restituisce la stringa segreta e un errore se l'operazione non è riuscita.

GetSecretStringWithStage

`func (c *Cache) GetSecretStringWithStage(secretId string, versionStage string) (string, error)`

GetSecretStringWithStage ottiene il valore della stringa segreta dalla cache per un determinato ID segreto e fase della versione. Restituisce la stringa segreta e un errore se l'operazione non è riuscita.

GetSecretBinary

`func (c *Cache) GetSecretBinary(secretId string) ([]byte, error) {`

GetSecretBinary ottiene il valore binario segreto dalla cache per un determinato ID segreto. Restituisce il numero binario del segreto e un errore in caso di errore dell'operazione.

GetSecretBinaryWithStage

```
func (c *Cache) GetSecretBinaryWithStage(secretId string, versionStage
string) ([]byte, error)
```

GetSecretBinaryWithStage ottiene il valore binario segreto dalla cache per un determinato ID segreto e fase della versione. Restituisce il numero binario del segreto e un errore in caso di errore dell'operazione.

tipo CacheConfig

Opzioni di configurazione della cache per una Cache, ad esempio dimensione massima della cache, fase della versione di default e durata (TTL) per i segreti memorizzati nella 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
}
```

tipo CacheHook

Un'interfaccia per collegarsi a una Cache per eseguire operazioni sul segreto memorizzato al suo interno.

Metodi

Put

```
Put(data interface{}) interface{}
```

Prepara l'oggetto per la memorizzazione nella cache.

Get

`Get(data interface{}) interface{}`

Deriva l'oggetto dall'oggetto memorizzato nella cache.

Ottieni un valore segreto di Secrets Manager utilizzando Go AWS SDK

Nelle applicazioni, puoi recuperare i tuoi segreti chiamando `GetSecretValue` o `BatchGetSecretValue` in uno qualsiasi dei. AWS SDKs Tuttavia, ti consigliamo di memorizzare nella cache i valori del segreto utilizzando la caching lato client. Memorizzare i segreti nella cache migliora la velocità e riduce i costi.

Per le applicazioni Go, utilizza il componente di caching basato su Go di Secrets Manager o chiama direttamente l'SDK con `GetSecretValue` o `BatchGetSecretValue`.

I seguenti esempi di codice mostrano come recuperare un valore segreto di Gestione dei segreti.

Autorizzazioni richieste:`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.
  }
```

Ottieni un valore segreto di Secrets Manager usando Rust

Nelle applicazioni, puoi recuperare i tuoi segreti chiamando `GetSecretValue` o `BatchGetSecretValue` in uno qualsiasi dei AWS SDKs. Tuttavia, ti consigliamo di memorizzare nella cache i valori del segreto utilizzando la caching lato client. Memorizzare i segreti nella cache migliora la velocità e riduce i costi.

Argomenti

- Ottieni un valore segreto di Secrets Manager usando Rust con la memorizzazione nella cache lato client
- Ottieni un valore segreto di Secrets Manager utilizzando Rust AWS SDK

Ottieni un valore segreto di Secrets Manager usando Rust con la memorizzazione nella cache lato client

Quando recuperi un segreto, puoi utilizzare il componente di caching basato su Secrets Manager Rust per memorizzarlo nella cache per usi futuri. Il recupero di un segreto memorizzato nella cache è più veloce rispetto al recupero da Secrets Manager. Poiché la chiamata a Secrets Manager comporta un costo APIs, l'utilizzo di una cache può ridurre i costi. Per tutti i modi in cui puoi recuperare i segreti, vedi Ottieni segreti.

La politica della cache è First In First Out (FIFO), quindi quando la cache deve eliminare un segreto, scarta il segreto più vecchio. Di default, la cache aggiorna i segreti ogni ora. È possibile configurare quanto segue:

- `max_size`— Il numero massimo di segreti memorizzati nella cache da mantenere prima di rimuovere i segreti a cui non è stato effettuato l'accesso di recente.
- `ttl`— La durata di validità di un elemento memorizzato nella cache prima di richiedere un aggiornamento dello stato segreto.

L'implementazione della cache non include l'invalidazione della cache. L'implementazione della cache è incentrata sulla cache stessa e non è rafforzata o focalizzata sulla sicurezza. Se hai bisogno di una sicurezza aggiuntiva, come la crittografia degli elementi nella cache, usa le caratteristiche fornite per modificare la cache.

Per utilizzare il componente, è necessario disporre di un ambiente di sviluppo Rust 2021 con. `tokio` Per ulteriori informazioni, consulta Guida introduttiva sul sito Web del linguaggio di programmazione Rust.

Per scaricare il codice sorgente, vedete il componente client di caching basato su Rust Secrets Manager su. GitHub

Per installare il componente di memorizzazione nella cache, utilizzare il comando seguente.

```
cargo add aws_secretsmanager_caching
```

Autorizzazioni richieste:

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

Per ulteriori informazioni, consulta Riferimento per le autorizzazioni.

Example Recupero di un segreto

L'esempio seguente mostra come ottenere il valore segreto per un segreto denominato*MyTest*.

```
use aws_secretsmanager_caching::SecretsManagerCachingClient;
use std::num::NonZeroUsize;
use std::time::Duration;

let client = match SecretsManagerCachingClient::default(
    NonZeroUsize::new(10).unwrap(),
    Duration::from_secs(60),
)
.await
{
    Ok(c) => c,
    Err(_) => panic!("Handle this error"),
};

let secret_string = match client.get_secret_value("MyTest", None, None).await {
    Ok(s) => s.secret_string.unwrap(),
    Err(_) => panic!("Handle this error"),
};

// Your code here
```

Example Istanziazione della cache con una configurazione personalizzata e un client personalizzato

L'esempio seguente mostra come configurare la cache e quindi ottenere il valore segreto per un segreto denominato. *MyTest*

```
let config = aws_config::load_defaults(BehaviorVersion::latest())
    .await
    .into_builder()
    .region(Region::from_static("us-west-2"))
    .build();

let asm_builder = aws_sdk_secretsmanager::config::Builder::from(&config);

let client = match SecretsManagerCachingClient::from_builder(
    asm_builder,
    NonZeroUsize::new(10).unwrap(),
```

```
    Duration::from_secs(60),
)
.await
{
    Ok(c) => c,
    Err(_) => panic!("Handle this error"),
};

let secret_string = client
    .get_secret_value("MyTest", None, None)
    .await
    {
        Ok(c) => c.secret_string.unwrap(),
        Err(_) => panic!("Handle this error"),
    };

// Your code here
```

Ottieni un valore segreto di Secrets Manager utilizzando Rust AWS SDK

Nelle applicazioni, puoi recuperare i tuoi segreti chiamando `GetSecretValue` o `BatchGetSecretValue` in uno qualsiasi dei. AWS SDKs Tuttavia, ti consigliamo di memorizzare nella cache i valori del segreto utilizzando la caching lato client. Memorizzare i segreti nella cache migliora la velocità e riduce i costi.

Per le applicazioni Rust, usa il componente di caching basato su Secrets Manager Rust o chiama l'SDK direttamente con o. GetSecretValue BatchGetSecretValue

I seguenti esempi di codice mostrano come recuperare un valore segreto di Gestione dei segreti.

Autorizzazioni richieste:`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(())
}
```

Usa Gestione dei segreti AWS i segreti in Amazon Elastic Kubernetes Service

Per mostrare i segreti di Gestione dei segreti AWS (ASCP) come file montati in Amazon EKS Pods, puoi utilizzare AWS Secrets and Configuration Provider per il driver CSI Kubernetes Secrets Store. L'ASCP funziona con Amazon Elastic Kubernetes Service 1.17+ che esegue un gruppo di nodi Amazon EC2. AWS Fargate i gruppi di nodi non sono supportati. Con ASCP, è possibile archiviare e gestire i segreti in Secrets Manager e recuperarli tramite i carichi di lavoro in esecuzione su Amazon EKS. Se il tuo segreto contiene più coppie chiave-valore in formato JSON, puoi scegliere quali montare in Amazon EKS. L'ASCP utilizza la sintassi JMESPath per interrogare le coppie chiave-valore nel tuo segreto. L'ASCP funziona anche con i parametri dell'archivio parametri. L'ASCP offre due metodi di autenticazione con Amazon EKS. Il primo approccio utilizza i ruoli IAM per gli account di servizio (IRSA). Il secondo approccio utilizza Pod Identity. Ogni approccio ha i suoi vantaggi e casi d'uso.

ASCP con ruoli IAM per gli account di servizio (IRSA)

L'ASCP con IAM Roles for Service Accounts (IRSA) ti consente di montare segreti da file Gestione dei segreti AWS as nei tuoi Amazon EKS Pods. Questo approccio è adatto quando:

- Devi montare i segreti come file nei tuoi Pods.
- Stai usando la versione 1.17 o quella successiva di Amazon EKS con i gruppi di nodi di Amazon EC2.
- Vuoi recuperare coppie chiave-valore specifiche da segreti in formato JSON.

Per ulteriori informazioni, consulta the section called “Integra ASCP con Pod Identity per Amazon EKS”.

ASCP con Pod Identity

ASCP con EKS Pod Identity

Il metodo ASCP con Pod Identity migliora la sicurezza e semplifica la configurazione per l'accesso ai segreti in Amazon EKS. Questo approccio è utile quando:

- È necessaria una gestione delle autorizzazioni più granulare a livello di pod.
- Stai utilizzando la versione 1.24 o quella successiva di Amazon EKS.

- Desideri prestazioni e scalabilità migliorate.

Per ulteriori informazioni, consulta the section called "Integra ASCP con Pod Identity per Amazon EKS".

Scelta dell'approccio giusto

Considera i seguenti fattori al momento di decidere tra ASCP con IRSA e ASCP con Pod Identity:

- Amazon EKSversion: Pod Identity richiede Amazon EKS 1.24+, mentre il driver CSI funziona con Amazon EKS 1.17+.
- Requisiti di sicurezza: Pod Identity offre un controllo più granulare a livello di pod.
- Prestazioni: Pod Identity generalmente offre prestazioni migliori in ambienti su larga scala.
- Complessità: Pod Identity semplifica la configurazione eliminando la necessità di account di servizio separati.

Scegli il metodo più adatto ai tuoi requisiti specifici e all'ambiente Amazon EKS.

Installa ASCP per Amazon EKS

Questa sezione spiega come installare AWS Secrets and Configuration Provider per Amazon EKS. Con ASCP, puoi montare segreti da Secrets Manager e parametri da AWS Systems Manager file in Amazon EKS Pods.

Prerequisiti

- Cluster Amazon EKS
  - Versione 1.24 o successiva per Pod Identity
  - Versione 1.17 o successiva per IRSA
- AWS CLI Installato e configurato
- kubectl installato e configurato per il cluster Amazon EKS
- Helm (versione 3.0 o successiva)

Installa e configura l'ASCP

L'ASCP è disponibile GitHub nel repository secrets-store-csi-provider-aws. Il repository contiene anche file YAML di esempio per la creazione e il montaggio di un segreto.

Durante l'installazione, puoi configurare l'ASCP per utilizzare un endpoint FIPS. Per un elenco di endpoint, consulta the section called "Endpoint di Secrets Manager".

Per installare ASCP come componente aggiuntivo EKS

1. Installazione `eksctl` (istruzioni di installazione)
2. Eseguite il comando seguente per installare il componente aggiuntivo con la configurazione predefinita:

   ```
   eksctl create addon --cluster <your_cluster> --name aws-secrets-store-csi-driver-provider
   ```

   Se desideri configurare il componente aggiuntivo, esegui invece il seguente comando di installazione:

   ```
   aws eks create-addon --cluster-name <your_cluster> --addon-name aws-secrets-store-csi-driver-provider --configuration-values 'file://path/to/config.yaml'
   ```

   Il file di configurazione può essere un file YAML o JSON. Per visualizzare lo schema di configurazione del componente aggiuntivo:

   a. Eseguite il comando seguente e prendete nota della versione più recente del componente aggiuntivo:

      ```
      aws eks describe-addon-versions --addon-name aws-secrets-store-csi-driver-provider
      ```

   b. Esegui il comando seguente per visualizzare lo schema di configurazione del componente aggiuntivo, sostituendolo `<version>` con la versione del passaggio precedente:

      ```
      aws eks describe-addon-configuration --addon-name aws-secrets-store-csi-driver-provider --addon-version <version>
      ```

Installazione dell'ASCP con l'utilizzo di Helm

1. Per assicurarti che il repository punti al grafico più recente, utilizza `helm repo update.`
2. Installare il grafico. Di seguito è riportato un esempio del `helm install` comando:

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

   a. Per utilizzare un endpoint FIPS, aggiungi il seguente flag: `--set useFipsEndpoint=true`
   b. Per configurare la limitazione (della larghezza di banda della rete), aggiungi il seguente flag: `--set-json 'k8sThrottlingParams={"qps": "`*`number of queries per second`*`", "burst": "`*`number of queries per second`*`"}'`
   c. Se il driver CSI Secrets Store è già installato sul cluster, aggiungi il seguente flag:`--set secrets-store-csi-driver.install=false`. Ciò eviterà l'installazione del driver CSI di Secrets Store come dipendenza.

Installazione utilizzando lo YAML nel repository

- Utilizza i seguenti comandi:

```
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
```

Verifica le installazioni

Per verificare le installazioni del cluster EKS, del driver CSI di Secrets Store e del plug-in ASCP, procedi nel seguente modo:

1. Verifica il cluster EKS:

```
eksctl get cluster --name clusterName
```

Questo comando deve restituire informazioni sul cluster.

2. Verifica l'installazione del driver CSI di Secrets Store:

```
kubectl get pods -n kube-system -l app=secrets-store-csi-driver
```

Dovresti vedere dei pod in esecuzione con nomi come `csi-secrets-store-secrets-store-csi-driver-xxx`.

3. Verifica l'installazione del plugin ASCP:

YAML installation

```
$ kubectl get pods -n kube-system -l app=csi-secrets-store-provider-aws
```

Output di esempio:

```
NAME                                     READY   STATUS    RESTARTS   AGE
csi-secrets-store-provider-aws-12345      1/1     Running   0          2m
```

Helm installation

```
$  kubectl get pods -n kube-system -l app=secrets-store-csi-driver-provider-aws
```

Output di esempio:

```
NAME                                                READY   STATUS    RESTARTS
 AGE
secrets-provider-aws-secrets-store-csi-driver-provider-67890       1/1
 Running   0          2m
```

Dovresti vedere dei pod nello stato `Running`.

Dopo aver eseguito questi comandi, se tutto è impostato correttamente, dovresti vedere tutti i componenti in esecuzione senza errori. In caso di problemi, potrebbe essere necessario risolverli controllando i log dei pod specifici che presentano problemi.

Risoluzione dei problemi

1. Per controllare i log del provider ASCP, esegui:

   ```
   kubectl logs -n kube-system -l app=csi-secrets-store-provider-aws
   ```

2. Controlla lo stato di tutti i pod nel namespace: `kube-system`

   ```
   kubectl -n kube-system get pods
   ```

   ```
   kubectl -n kube-system logs pod/PODID
   ```

   Tutti i pod relativi al driver CSI e all'ASCP devono essere nello stato "In esecuzione".

3. Controlla la versione del driver CSI:

   ```
   kubectl get csidriver secrets-store.csi.k8s.io -o yaml
   ```

   Questo comando deve restituire informazioni sul driver CSI installato.

Risorse aggiuntive

Per ulteriori informazioni su come utilizzare ASCP con Amazon EKS, consulta le seguenti risorse:

- Utilizzo di Pod Identity con Amazon EKS
- AWS Il driver CSI di Secrets Store è attivo GitHub

Usa AWS Secrets e Configuration Provider CSI con Pod Identity per Amazon EKS

L'integrazione di AWS Secrets and Configuration Provider con Pod Identity Agent per Amazon Elastic Kubernetes Service offre maggiore sicurezza, configurazione semplificata e prestazioni migliorate per le applicazioni in esecuzione su Amazon EKS. Pod Identity semplifica l'autenticazione IAM per Amazon EKS durante il recupero di segreti da Secrets Manager o parametri da AWS Systems Manager Parameter Store.

Amazon EKS Pod Identity semplifica il processo di configurazione delle autorizzazioni IAM per le applicazioni Kubernetes consentendo quest'azione direttamente tramite le interfacce Amazon EKS,

riducendo il numero di passaggi ed eliminando la necessità di passare da Amazon EKS ai servizi IAM. Pod Identity consente l'uso di un singolo ruolo IAM su più cluster senza aggiornare le policy di attendibilità e supporta i tag delle sessioni di ruolo per un controllo degli accessi più granulare. Questo approccio non solo semplifica la gestione delle policy consentendo il riutilizzo delle politiche di autorizzazione tra i ruoli, ma migliora anche la sicurezza abilitando l'accesso alle risorse in base ai AWS tag corrispondenti.

Come funziona

1. Pod Identity assegna un ruolo IAM al pod.
2. ASCP utilizza questo ruolo per l'autenticazione con. Servizi AWS
3. Se autorizzato, ASCP recupera i segreti richiesti e li rende disponibili al Pod.

Per ulteriori informazioni, consulta Scopri come funziona Amazon EKS Pod Identity nella Guida per l'utente di Amazon EKS.

Prerequisiti

> **Important**
>
> Pod Identity è supportato solo per Amazon EKS nel cloud. Non è supportato per Amazon EKS Anywhere o per i cluster Servizio Red Hat OpenShift su AWSKubernetes autogestiti sulle istanze Amazon EC2.

- Cluster Amazon EKS (versione 1.24 o successiva)
- Accesso AWS CLI e cluster Amazon EKS tramite `kubectl`
- Accesso a due Account AWS (per l'accesso su più account)

Installazione dell'agente Amazon EKS Pod Identity

Per utilizzare Pod Identity con il tuo cluster, devi installare il componente aggiuntivo dell'agente Amazon EKS Pod Identity.

Installare l'agente Pod Identity

- Installa il componente aggiuntivo Pod Identity Agent sul tuo cluster:

```
eksctl create addon \
  --name eks-pod-identity-agent \
  --cluster clusterName \
  --region region
```

Configura ASCP con Pod Identity

1. Crea una politica di autorizzazioni che conceda `secretsmanager:GetSecretValue` e `secretsmanager:DescribeSecret` autorizzi i segreti a cui il Pod deve accedere. Per un esempio di policy, consulta the section called "Esempio: autorizzazione a leggere e descrivere singoli segreti".
2. Crea un ruolo IAM che può essere assunto dal servizio principale Amazon EKS per Pod Identity:

   JSON

   ```
   {
       "Version":"2012-10-17",
       "Statement": [
         {
           "Effect": "Allow",
           "Principal": {
             "Service": "pods.eks.amazonaws.com"
           },
           "Action": [
             "sts:AssumeRole",
             "sts:TagSession"
           ]
         }
       ]
     }
   ```

   Allega la policy IAM al ruolo:

   ```
   aws iam attach-role-policy \
     --role-name MY_ROLE \
     --policy-arn POLICY_ARN
   ```

3. Crea un'associazione Pod Identity. Per un esempio, consulta Crea un'associazione Pod Identity nella Guida per l'utente Amazon EKS
4. Crea il `SecretProviderClass` file che specifica quali segreti montare nel Pod:

   ```
   kubectl apply -f https://raw.githubusercontent.com/aws/secrets-store-csi-driver-provider-aws/main/examples/ExampleSecretProviderClass-PodIdentity.yaml
   ```

   La differenza fondamentale in `SecretProviderClass` tra IRSA e Pod Identity è il parametro `usePodIdentity` facoltativo. Si tratta di un campo facoltativo che determina l'approccio di autenticazione. Quando non è specificato, per impostazione predefinita utilizzare i ruoli IAM per gli account di servizio (IRSA).

   - Per utilizzare EKS Pod Identity, utilizza uno qualsiasi di questi valori: `"true", "True", "TRUE", "t", "T"`.
   - Per utilizzare in modo esplicito IRSA, imposta uno qualsiasi di questi valori: `"false", "False", "FALSE", "f", or "F"`.
5. Implementa il Pod che monta i segreti sotto: `/mnt/secrets-store`

   ```
   kubectl apply -f https://raw.githubusercontent.com/aws/secrets-store-csi-driver-provider-aws/main/examples/ExampleDeployment-PodIdentity.yaml
   ```
6. Se utilizzi un cluster Amazon EKS privato, assicurati che il VPC in cui si trova il cluster abbia un AWS STS endpoint. Per informazioni sulla creazione di un endpoint, consulta Endpoint VPC di interfaccia nella Guida per l'utente AWS Identity and Access Management .

Verifica il montaggio del segreto

Per verificare che il segreto sia montato correttamente, esegui il seguente comando:

```
kubectl exec -it $(kubectl get pods | awk '/pod-identity-deployment/{print $1}' | head -1) -- cat /mnt/secrets-store/MySecret
```

Per configurare Amazon EKS Pod Identity per l'accesso ai segreti in Secrets Manager

1. Crea una politica di autorizzazioni che conceda `secretsmanager:GetSecretValue` e `secretsmanager:DescribeSecret` autorizzi i segreti a cui il Pod deve accedere. Per un esempio di policy, consulta the section called “Esempio: autorizzazione a leggere e descrivere singoli segreti”.

2. Crea un segreto in Secrets Manager, se non ne hai già uno.

Risoluzione dei problemi

È possibile visualizzare la maggior parte degli errori descrivendo l'implementazione del pod.

Per visualizzare i messaggi di errore per il container

1. È possibile ottenere un elenco di nomi di pod con il comando seguente. Se non si sta utilizzando il namespace predefinito, utilizzare `-n` *`NAMESPACE`*.

   ```
   kubectl get pods
   ```

2. Per descrivere il Pod, nel comando seguente, *`PODID`* usa l'ID Pod dei Pod che hai trovato nel passaggio precedente. Se non si sta utilizzando lo spazio dei nomi predefinito, utilizzare `-n` *`NAMESPACE`*.

   ```
   kubectl describe pod/PODID
   ```

Come visualizzare gli errori per l'ASCP

- Per trovare maggiori informazioni nei log del provider, nel comando seguente, *`PODID`* usa l'ID del Pod csi-secrets-store-provider-aws.

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

Usa AWS Secrets and Configuration Provider CSI con IAM Roles for Service Accounts (IRSA)

Argomenti

Prerequisiti

- Cluster Amazon EKS (versione 1.17 o successiva)
- Accesso AWS CLI e cluster Amazon EKS tramite `kubectl`

Configurazione del controllo degli accessi

L'ASCP recupera il Pod Identity di Amazon EKS e lo scambia con il ruolo IAM. Le autorizzazioni vengono impostate in una policy IAM per quel ruolo IAM. Quando ASCP assume il ruolo IAM, ottiene l'accesso ai segreti che hai autorizzato. Altri container non possono accedere ai segreti a meno che non vengano associati anche al ruolo IAM.

Per consentire al tuo Amazon EKS Pod di accedere ai segreti in Secrets Manager

1. Crea una politica di autorizzazioni che conceda `secretsmanager:GetSecretValue` e `secretsmanager:DescribeSecret` autorizzi i segreti a cui il Pod deve accedere. Per un esempio di policy, consulta the section called "Esempio: autorizzazione a leggere e descrivere singoli segreti".
2. Crea un provider OpenID Connect (OIDC) IAM per il cluster se non ne è già presente uno. Per ulteriori informazioni, consulta Crea un provider IAM OIDC per il tuo cluster nella Guida per l'utente di Amazon EKS.
3. Crea un Ruolo IAM per l'account del servizio e collega la policy ad esso. Per ulteriori informazioni, consulta Crea un ruolo IAM per un account del servizio nella Guida per l'utente di Amazon EKS.
4. Se utilizzi un cluster Amazon EKS privato, assicurati che il VPC in cui si trova il cluster abbia un AWS STS endpoint. Per informazioni sulla creazione di un endpoint, consulta Endpoint VPC di interfaccia nella Guida per l'utente AWS Identity and Access Management .

Identificazione dei segreti montare

Per determinare quali segreti l'ASCP monta in Amazon EKS come file sul file system, è necessario creare un file the section called "SecretProviderClass" YAML. `SecretProviderClass`Elenca i segreti da montare e il nome del file con cui montarli. Il `SecretProviderClass` deve trovarsi nello stesso namespace del pod Amazon EKS a cui fa riferimento.

Monta i segreti come file

Le seguenti istruzioni mostrano come montare i segreti come file utilizzando file YAML di esempio .yaml e ExampleSecretProviderClass.yaml. ExampleDeployment

Per montare segreti in Amazon EKS

1. Applica il `SecretProviderClass` al pod:

```
kubectl apply -f ExampleSecretProviderClass.yaml
```

2. Distribuisci il pod:

```
kubectl apply -f ExampleDeployment.yaml
```

3. L'ASCP monta i file.

Risoluzione dei problemi

È possibile visualizzare la maggior parte degli errori descrivendo l'implementazione del pod.

Per visualizzare i messaggi di errore per il container

1. È possibile ottenere un elenco di nomi di pod con il comando seguente. Se non si sta utilizzando il namespace predefinito, utilizzare `-n` *`nameSpace`*.

```
kubectl get pods
```

2. Per descrivere il Pod, nel comando seguente, *`podId`* usa l'ID Pod dei Pod che hai trovato nel passaggio precedente. Se non si sta utilizzando lo spazio dei nomi predefinito, utilizzare `-n` *`nameSpace`*.

```
kubectl describe pod/podId
```

Come visualizzare gli errori per l'ASCP

- Per trovare maggiori informazioni nei log del provider, nel comando seguente, *`podId`* usa l'ID del Pod csi-secrets-store-provider-aws.

```
kubectl -n kube-system get pods
```

```
kubectl -n kube-system logs Pod/podId
```

- Verifica che il **`SecretProviderClass`** sia installato:

  ```
  kubectl get crd secretproviderclasses.secrets-store.csi.x-k8s.io
  ```

  Questo comando deve restituire informazioni sulla definizione delle risorse personalizzate `SecretProviderClass`.

- Verifica che l' SecretProviderClass oggetto sia stato creato.

  ```
  kubectl get secretproviderclass SecretProviderClassName -o yaml
  ```

AWS Esempi di codice Secrets and Configuration Provider

Esempi di autenticazione e controllo degli accessi ASCP

Esempio: policy IAM che consente al servizio Amazon EKS Pod Identity (pods.eks.amazonaws.com) di assumere il ruolo e contrassegnare la sessione:

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "pods.eks.amazonaws.com"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:TagSession"
      ]
    }
  ]
}
```

SecretProviderClass

Puoi utilizzare YAML per descrivere quali segreti montare in Amazon EKS utilizzando l'ASCP. Per alcuni esempi, consulta the section called "SecretProviderClass utilizzo".

SecretProviderClass Struttura YAML

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

Il campo parametri contiene i dettagli della richiesta di montaggio:

region

(Facoltativo) Il Regione AWS segreto. Se non utilizzi questo campo, l'ASCP cerca la Regione dall'annotazione sul nodo. Questa ricerca aggiunge un sovraccarico alle richieste di montaggio, quindi consigliamo di fornire la Regione per i cluster che utilizzano un numero elevato di pod.

Se specifichi anche la `failoverRegion`, l'ASCP tenta di recuperare il segreto da entrambe le Regioni. Se una delle Regioni restituisce un errore 4xx, ad esempio per un problema di autenticazione, l'ASCP non installa nessuno dei due segreti. Se il segreto viene recuperato correttamente da `region`, l'ASCP monta tale valore del segreto. Se il segreto non viene recuperato correttamente da `region`, ma viene correttamente recuperato da `failoverRegion`, l'ASCP monta tale valore del segreto.

failoverRegion

(Facoltativo) Se si include questo campo, l'ASCP tenta di recuperare il segreto dalle Regioni definite in`region` e in questo campo. Se una delle Regioni restituisce un errore 4xx, ad esempio per un problema di autenticazione, l'ASCP non installa nessuno dei due segreti. Se il

segreto viene recuperato correttamente da `region`, l'ASCP monta tale valore del segreto. Se il segreto non viene recuperato correttamente da `region`, ma viene correttamente recuperato da `failoverRegion`, l'ASCP monta tale valore del segreto. Per un esempio su come utilizzare questo campo, consulta la sezione Failover segreto in più regioni.

pathTranslation

(Facoltativo) Un singolo carattere di sostituzione da utilizzare se il nome del file in Amazon EKS conterrà il carattere separatore di percorso, come barra (/) su Linux. L'ASCP non è in grado di creare un file montato che contiene un carattere separatore di percorso. Invece, l'ASCP sostituisce il carattere separatore di percorso con un carattere diverso. Se non utilizzi questo campo, il carattere sostitutivo è il carattere di sottolineatura (_), quindi ad esempio, `My/Path/Secret` monta come `My_Path_Secret`.

Per impedire la sostituzione dei caratteri, immettere la stringa `False`.

usePodIdentity

(Facoltativo) Determina l'approccio di autenticazione. Se non specificato, il valore predefinito è ruoli IAM per gli account di servizio (IRSA).

- Per utilizzare EKS Pod Identity, utilizza uno qualsiasi di questi valori: `"true""`, `"True"`, `"TRUE"`, `"t"` o `"T"`.
- Per utilizzare in modo esplicito IRSA, imposta uno qualsiasi dei seguenti valori: `"false"`, `"False"`, `"FALSE"`, `"f"`, or `"F""=`.

preferredAddressType

(Facoltativo) Specifica il tipo di indirizzo IP preferito per la comunicazione con gli endpoint dell'agente Pod Identity. Il campo è applicabile solo quando si utilizza la funzionalità EKS Pod Identity e verrà ignorato, quando si utilizzano i ruoli IAM per gli account di servizio. I valori non fanno distinzione tra maiuscole e minuscole. I valori validi sono:

- `"ipv4"`, `"IPv4"` «, oppure `"IPV4"` — Forza l'uso dell' IPv4 endpoint Pod Identity Agent
- `"ipv6"`,`"IPv6"`, o `"IPV6"` — Forza l'uso dell'endpoint Pod Identity Agent IPv6
- non specificato: utilizza la selezione automatica degli endpoint, provando prima l' IPv4 endpoint e ritornando all' IPv6 endpoint se fallisce IPv4

objects

Una stringa contenente una dichiarazione YAML dei segreti da montare. Si consiglia di utilizzare una stringa multiriga YAML o un carattere pipe (|).

objectName

Obbligatorio. Speciifica il nome del segreto o del parametro da recuperare. Per Secrets Manager questo è il parametro `SecretId` e può essere sia il nome descrittivo sia l'ARN completo del segreto. Per SSM Parameter Store, questo è il `Name`parametro e può essere il nome o l'ARN completo del parametro.

objectType

Obbligatorio se non si utilizza un ARN di Secrets Manager per `objectName`. Può essere `secretsmanager` o `ssmparameter`.

objectAlias

(Facoltativo) Il nome file del segreto nel pod Amazon EKS. Se non indichi questo campo, `objectName` viene visualizzato come nome del file.

Autorizzazione del file

(Facoltativo) La stringa ottale a 4 cifre che specifica l'autorizzazione del file con cui eseguire il montaggio segreto. Se non specifichi questo campo, il valore predefinito sarà. `"0644"`

objectVersion

(Facoltativo) L'ID di versione del segreto. Non è consigliato perché è necessario aggiornare l'ID di versione ogni volta che si aggiorna il segreto. Per impostazione predefinita viene utilizzata la versione più recente. Se includi un `failoverRegion`, questo campo rappresenta l'`objectVersion` primario.

objectVersionLabel

(Facoltativo) L'alias per la versione. Il valore predefinito è la versione più recente AWSCURRENT. Per ulteriori informazioni, consulta the section called "Versioni segrete". Se includi un `failoverRegion`, questo campo rappresenta l'`objectVersionLabel` primario.

jmesPath

(Facoltativo) Una mappa delle chiavi nel segreto per i file da montare in Amazon EKS. Per utilizzare questo campo, il valore del segreto deve essere in formato JSON. Se si utilizza questo campo, è necessario includere i sottocampi `path` e `objectAlias`.

path

Una chiave da una coppia chiave-valore nel JSON del valore segreto. Se il campo contiene un trattino, usa le virgolette singole per evitarlo, ad esempio: `path: '"hyphenated-path"'`

objectAlias

Il nome del file da montare nel pod Amazon EKS. Se il campo contiene un trattino, usa le virgolette singole per evitarlo, ad esempio: `objectAlias: '"hyphenated-alias"'`

Autorizzazione del file

(Facoltativo) La stringa ottale a 4 cifre che specifica l'autorizzazione del file con cui eseguire il montaggio segreto. Se non si specifica questo campo, verrà utilizzata per impostazione predefinita l'autorizzazione al file dell'oggetto principale.

failoverObject

(Facoltativo) Se specifichi questo campo, l'ASCP tenta di recuperare sia il segreto specificato nel campo primario `objectName` che il segreto specificato nel sottocampo `failoverObject` `objectName`. Se uno dei due restituisce un errore 4xx, ad esempio per un problema di autenticazione, l'ASCP non installa nessuno dei due segreti. Se il segreto viene recuperato con successo dall'`objectName` primario, l'ASCP monta tale valore del segreto. Se il segreto non viene recuperato correttamente dall'`objectName` primario, ma viene recuperato con successo dall'`objectName` di failover, l'ASCP installa tale valore del segreto. Se si include questo campo, è necessario includere il campo `objectAlias`. Per un esempio su come utilizzare questo campo, consulta la sezione Failover su un segreto diverso.

In genere, si utilizza questo campo quando il segreto di failover non è una replica. Per un esempio su come specificare una replica, consulta la sezione Failover segreto in più regioni.

objectName

Il nome o l'ARN completo del segreto di failover. Se utilizzi un ARN, la Regione nell'ARN deve corrispondere al campo `failoverRegion`.

objectVersion

(Facoltativo) L'ID di versione del segreto. Deve corrispondere all'`objectVersion` primaria. Non è consigliato perché è necessario aggiornare l'ID di versione ogni volta che si aggiorna il segreto. Per impostazione predefinita viene utilizzata la versione più recente.

objectVersionLabel

(Facoltativo) L'alias per la versione. L'impostazione predefinita è la versione più recente AWSCURRENT. Per ulteriori informazioni, consulta the section called "Versioni segrete".

Crea una SecretProviderClass configurazione di base per montare i segreti nei tuoi Amazon EKS Pods.

Pod Identity

SecretProviderClass per utilizzare un segreto nello stesso cluster Amazon EKS:

```
apiVersion: secrets-store.csi.x-k8s.io/v1
kind: SecretProviderClass
metadata:
  name: aws-secrets-manager
spec:
  provider: aws
  parameters:
    objects: |
      - objectName: "mySecret"
        objectType: "secretsmanager"
    usePodIdentity: "true"
```

IRSA

```
apiVersion: secrets-store.csi.x-k8s.io/v1
kind: SecretProviderClass
metadata:
  name: deployment-aws-secrets
spec:
  provider: aws
  parameters:
    objects: |
        - objectName: "MySecret"
          objectType: "secretsmanager"
```

SecretProviderClass utilizzo

Usa questi esempi per creare SecretProviderClass configurazioni per diversi scenari.

Esempio: montaggio di segreti per nome o ARN

Questo esempio mostra come montare tre diversi tipi di segreti:

- Un segreto specificato dall'ARN completo
- Un segreto specificato per nome

- Una versione specifica di un segreto

```
apiVersion: secrets-store.csi.x-k8s.io/v1
kind: SecretProviderClass
metadata:
  name: aws-secrets
spec:
  provider: aws
  parameters:
    objects: |
      - objectName: "arn:aws:secretsmanager:us-east-2:777788889999:secret:MySecret2-
d4e5f6"
      - objectName: "MySecret3"
        objectType: "secretsmanager"
      - objectName: "MySecret4"
        objectType: "secretsmanager"
        objectVersionLabel: "AWSCURRENT"
```

Esempio: monta coppie chiave-valore da un segreto

Questo esempio mostra come montare coppie chiave-valore specifiche da un segreto in formato JSON:

```
apiVersion: secrets-store.csi.x-k8s.io/v1
kind: SecretProviderClass
metadata:
  name: aws-secrets
spec:
  provider: aws
  parameters:
    objects: |
      - objectName: "arn:aws:secretsmanager:us-east-2:777788889999:secret:MySecret-
a1b2c3"
        jmesPath:
            - path: username
              objectAlias: dbusername
            - path: password
              objectAlias: dbpassword
```

Esempio: monta i segreti tramite l'autorizzazione del file

Questo esempio mostra come montare un segreto con un'autorizzazione specifica per il file

```
apiVersion: secrets-store.csi.x-k8s.io/v1
kind: SecretProviderClass
metadata:
  name: aws-secrets
spec:
  provider: aws
  parameters:
    objects: |
      - objectName: "mySecret"
        objectType: "secretsmanager"
        filePermission: "0600"
        jmesPath:
            - path: username
              objectAlias: dbusername
              filePermission: "0400"
```

Esempio: esempi di configurazione di failover

Questi esempi mostrano come configurare il failover per i segreti.

Failover segreto in più regioni

Questo esempio mostra come configurare il failover automatico per un segreto replicato su più regioni:

```
apiVersion: secrets-store.csi.x-k8s.io/v1
kind: SecretProviderClass
metadata:
  name: aws-secrets
spec:
  provider: aws
  parameters:
    region: us-east-1
    failoverRegion: us-east-2
    objects: |
      - objectName: "MySecret"
```

Failover su un segreto diverso

Questo esempio mostra come configurare il failover su un segreto diverso (non su una replica):

```
apiVersion: secrets-store.csi.x-k8s.io/v1
```

```
kind: SecretProviderClass
metadata:
  name: aws-secrets
spec:
  provider: aws
  parameters:
    region: us-east-1
    failoverRegion: us-east-2
    objects: |
      - objectName: "arn:aws:secretsmanager:us-east-1:777788889999:secret:MySecret-
a1b2c3"
        objectAlias: "MyMountedSecret"
        failoverObject:
          - objectName: "arn:aws:secretsmanager:us-
east-2:777788889999:secret:MyFailoverSecret-d4e5f6"
```

Risorse aggiuntive

Per ulteriori informazioni su come utilizzare ASCP con Amazon EKS, consulta le seguenti risorse:

- Utilizzo di Pod Identity con Amazon EKS
- Utilizzo di AWS Secrets and Configuration Provider
- AWS Secrets Store CSI Driver attivo GitHub

Usa Gestione dei segreti AWS i segreti nelle AWS Lambda funzioni

AWS Lambda è un servizio di elaborazione senza server che consente di eseguire codice senza effettuare il provisioning o la gestione dei server. Parameter Store, una funzionalità di AWS Systems Manager, fornisce uno storage sicuro e gerarchico per la gestione dei dati di configurazione e la gestione dei segreti. Puoi utilizzare l'estensione Lambda AWS Parameters and Secrets per recuperare e memorizzare nella cache Gestione dei segreti AWS i segreti e i parametri Parameter Store nelle funzioni Lambda senza utilizzare un SDK. Per informazioni dettagliate sull'uso di questa estensione, consulta Use Secrets Manager secret nelle funzioni Lambda nella Lambda Developer Guide.

Utilizzo dei segreti di Secrets Manager con Lambda

La Lambda Developer Guide fornisce istruzioni complete per l'utilizzo dei segreti di Secrets Manager nelle funzioni Lambda. Per iniziare

1. Segui il step-by-step tutorial in Use Secrets Manager secrets in Lambda functions, che include:
   - Creazione di una funzione Lambda con il runtime preferito (Python, Node.js, Java)
   - Aggiungere l'estensione Lambda AWS Parameters and Secrets come livello
   - Configurazione delle autorizzazioni necessarie
   - Scrittura di codice per recuperare i segreti dall'estensione
   - Verifica della tua funzione
2. Scopri le variabili di ambiente per configurare il comportamento dell'estensione, incluse le impostazioni della cache e i timeout
3. Comprendi le migliori pratiche per lavorare con la rotazione segreta

Utilizzo di Secrets Manager e Lambda in un VPC

Se la tua funzione Lambda viene eseguita in un VPC, devi creare un endpoint VPC in modo che l'estensione possa effettuare chiamate a Secrets Manager. Per ulteriori informazioni, consulta the section called "Endpoint VPC (AWS PrivateLink)".

Utilizzo dell'estensione Lambda AWS Parameters and Secrets

L'estensione può recuperare sia i segreti di Gestione dei segreti che i parametri di Archivio dei parametri Per informazioni dettagliate sull'utilizzo dei parametri Parameter Store con l'estensione, vedere Utilizzo dei parametri Parameter Store nelle funzioni Lambda nella Guida per l'AWS Systems Manager utente.

La documentazione di Systems Manager include:

- Spiegazione dettagliata del funzionamento dell'estensione con Parameter Store
- Istruzioni per aggiungere l'estensione a una funzione Lambda
- Variabili di ambiente per la configurazione dell'estensione
- Comandi di esempio per il recupero dei parametri
- Elenco completo delle estensioni ARNs per tutte le architetture e le regioni supportate

Utilizzo dell' Gestione dei segreti AWS agente

Come funziona l'agente Secrets Manager

L' Gestione dei segreti AWS Agent è un servizio HTTP lato client che ti aiuta a standardizzare il modo in cui utilizzi i segreti di Secrets Manager nei tuoi ambienti di elaborazione. È possibile utilizzarlo con i seguenti servizi:

- AWS Lambda
- Amazon Elastic Container Service
- Amazon Elastic Kubernetes Service
- Amazon Elastic Compute Cloud

L'agente Secrets Manager recupera e memorizza nella cache i segreti in memoria, consentendo alle applicazioni di ottenere segreti da localhost invece di effettuare chiamate dirette a Secrets Manager. L'agente Secrets Manager può solo leggere i segreti, non può modificarli.

> **Important**
>
> L'agente Secrets Manager utilizza AWS le credenziali dell'ambiente per chiamare Secrets Manager. Include la protezione contro Server Side Request Forgery (SSRF) per contribuire a migliorare la sicurezza segreta. Per impostazione predefinita, l'agente Secrets Manager utilizza lo scambio di chiavi ML-KEM post-quantistico come scambio di chiavi con la massima priorità.

Informazioni sulla memorizzazione nella cache dell'agente Secrets Manager

L'agente Secrets Manager utilizza una cache in memoria che si ripristina al riavvio dell'agente Secrets Manager. Aggiorna periodicamente i valori segreti memorizzati nella cache in base a quanto segue:

- La frequenza di aggiornamento predefinita (TTL) è 300 secondi
- È possibile modificare il TTL utilizzando un file di configurazione
- L'aggiornamento avviene quando si richiede un segreto dopo la scadenza del TTL

> **Note**
>
> L'agente Secrets Manager non include l'invalidazione della cache. Se un segreto ruota prima della scadenza della voce della cache, l'agente Secrets Manager potrebbe restituire un valore segreto non aggiornato.

L'agente Secrets Manager restituisce valori segreti nello stesso formato della risposta di`GetSecretValue`. I valori segreti non sono crittografati nella cache.

Argomenti

Crea l'agente Secrets Manager

Prima di iniziare, assicurati di avere gli strumenti di sviluppo standard e gli strumenti Rust installati per la tua piattaforma.

> **Note**
>
> La creazione dell'agente con la `fips` funzionalità abilitata su macOS richiede attualmente la seguente soluzione alternativa:
>
> - Crea una variabile di ambiente chiamata `SDKROOT` che è impostata sul risultato dell'esecuzione `xcrun --show-sdk-path`

RPM-based systems

Per creare su sistemi basati su RPM

1. Usa lo `install` script fornito nel repository.

   Lo script genera un token SSRF casuale all'avvio e lo memorizza nel file. `/var/run/awssmatoken` Il token è leggibile dal `awssmatokenreader` gruppo creato dallo script di installazione.

2. Per consentire all'applicazione di leggere il file del token, è necessario aggiungere al `awssmatokenreader` gruppo l'account utente con cui viene eseguita l'applicazione. Ad esempio, potete concedere all'applicazione le autorizzazioni per leggere il file del token con il seguente comando usermod, dove si *<APP_USER>* trova l'ID utente con cui viene eseguita l'applicazione.

   ```
   sudo usermod -aG awssmatokenreader <APP_USER>
   ```

   Installa gli strumenti di sviluppo

   Su sistemi basati su RPM, ad esempio AL2023, installa il gruppo Development Tools:

   ```
   sudo yum -y groupinstall "Development Tools"
   ```

3. Installa Rust

   Segui le istruzioni su Install Rust nella documentazione di Rust:

   ```
   curl --proto '=https' --tlsv1.2 -sSf https://sh.rustup.rs | sh # Follow the on-
   screen instructions
   . "$HOME/.cargo/env"
   ```

4. Crea l'agente

   Crea l'agente Secrets Manager usando il comando cargo build:

   ```
   cargo build --release
   ```

   Troverai l'eseguibile sotto`target/release/aws_secretsmanager_agent`.

Debian-based systems

Per costruire su sistemi basati su Debian

1. Installa strumenti di sviluppo

   Su sistemi basati su Debian come Ubuntu, installa il pacchetto build-essential:

   ```
   sudo apt install build-essential
   ```

2. Installa Rust

   Segui le istruzioni su Install Rust nella documentazione di Rust:

   ```
   curl --proto '=https' --tlsv1.2 -sSf https://sh.rustup.rs | sh # Follow the on-
   screen instructions
   . "$HOME/.cargo/env"
   ```

3. Crea l'agente

   Crea l'agente Secrets Manager usando il comando cargo build:

   ```
   cargo build --release
   ```

   Troverai l'eseguibile sotto`target/release/aws_secretsmanager_agent`.

Windows

Per creare su Windows

1. Configurare l'ambiente di sviluppo

   Segui le istruzioni in Configurare l'ambiente di sviluppo su Windows for Rust nella documentazione di Microsoft Windows.

2. Crea l'agente

   Crea l'agente Secrets Manager usando il comando cargo build:

   ```
   cargo build --release
   ```

Troverai l'eseguibile sotto`target/release/aws_secretsmanager_agent.exe`.

Cross-compile natively

Per eseguire la compilazione incrociata in modo nativo

1. Installa strumenti di compilazione incrociata

   Nelle distribuzioni in cui è disponibile il pacchetto mingw-w64 come Ubuntu, installa la toolchain cross-compile:

   ```
   # Install the cross compile tool chain
   sudo add-apt-repository universe
   sudo apt install -y mingw-w64
   ```

2. Aggiungi obiettivi di compilazione di Rust

   Installa il target di build di Windows GNU:

   ```
   rustup target add x86_64-pc-windows-gnu
   ```

3. Compila per Windows

   Compila in modo incrociato l'agente per Windows:

   ```
   cargo build --release --target x86_64-pc-windows-gnu
   ```

   L'eseguibile è disponibile all'indirizzo. `target/x86_64-pc-windows-gnu/release/aws_secretsmanager_agent.exe`

Cross compile with Rust cross

Per eseguire la compilazione incrociata utilizzando Rust cross

Se gli strumenti di compilazione incrociata non sono disponibili in modo nativo sul sistema, puoi utilizzare il progetto Rust cross. Per ulteriori informazioni, consulta cross. https://github.com/cross-rs/

> **Important**
>
> Consigliamo 32 GB di spazio su disco per l'ambiente di compilazione.

1. Configura Docker

   Installa e configura Docker:

   ```
   # Install and start docker
   sudo yum -y install docker
   sudo systemctl start docker
   sudo systemctl enable docker # Make docker start after reboot
   ```

2. Configura le autorizzazioni Docker

   Aggiungi il tuo utente al gruppo docker:

   ```
   # Give ourselves permission to run the docker images without sudo
   sudo usermod -aG docker $USER
   newgrp docker
   ```

3. Compila per Windows

   Installa cross e crea l'eseguibile:

   ```
   # Install cross and cross compile the executable
   cargo install cross
   cross build --release --target x86_64-pc-windows-gnu
   ```

Installare l'agente Secrets Manager

Scegli il tuo ambiente di elaborazione tra le seguenti opzioni di installazione.

Amazon EC2

Per installare l'agente Secrets Manager su Amazon EC2

1. Passa alla directory di configurazione

   Passa alla directory di configurazione:

```
cd aws_secretsmanager_agent/configuration
```

2. Esegui lo script di installazione

   Esegui lo `install` script fornito nel repository.

   Lo script genera un token SSRF casuale all'avvio e lo memorizza nel file. `/var/run/awssmatoken` Il token è leggibile dal `awssmatokenreader` gruppo creato dallo script di installazione.

3. Configura le autorizzazioni dell'applicazione

   Aggiungi l'account utente con cui viene eseguita l'applicazione al `awssmatokenreader` gruppo:

   ```
   sudo usermod -aG awssmatokenreader APP_USER
   ```

   Sostituiscilo *APP_USER* con l'ID utente con cui viene eseguita l'applicazione.

Container Sidecar

Puoi eseguire Secrets Manager Agent come contenitore secondario insieme all'applicazione utilizzando Docker. Quindi l'applicazione può recuperare i segreti dal server HTTP locale fornito dall'agente Secrets Manager. Per informazioni su Docker, consulta la documentazione Docker.

Per creare un contenitore sidecar per l'agente Secrets Manager

1. Crea l'agente Dockerfile

   Crea un Dockerfile per il contenitore sidecar di Secrets Manager Agent:

   ```
   # Use the latest Debian image as the base
   FROM debian:latest

   # Set the working directory inside the container
   WORKDIR /app

   # Copy the Secrets Manager Agent binary to the container
   COPY secrets-manager-agent .

   # Install any necessary dependencies
   ```

```
RUN apt-get update && apt-get install -y ca-certificates

# Set the entry point to run the Secrets Manager Agent binary
ENTRYPOINT ["./secrets-manager-agent"]
```

2. Crea l'applicazione Dockerfile

   Crea un Dockerfile per la tua applicazione client.

3. Crea il file Docker Compose

   Crea un file Docker Compose per eseguire entrambi i contenitori con un'interfaccia di rete condivisa:

> **Important**
>
> È necessario caricare AWS le credenziali e il token SSRF affinché l'applicazione possa utilizzare l'agente Secrets Manager. Per Amazon EKS e Amazon ECS, consulta quanto segue:
>
> - Gestisci l'accesso nella Guida per l'utente di Amazon EKS
> - Task Amazon ECS (ruolo IAM) nella Amazon ECS Developer Guide

```
version: '3'
services:
    client-application:
    container_name: client-application
    build:
        context: .
        dockerfile: Dockerfile.client
    command: tail -f /dev/null  # Keep the container running


    secrets-manager-agent:
    container_name: secrets-manager-agent
    build:
        context: .
        dockerfile: Dockerfile.agent
    network_mode: "container:client-application"  # Attach to the client-
application container's network
    depends_on:
```

```
        - client-application
```

4. File binario dell'agente di copia

   Copia il file `secrets-manager-agent` binario nella stessa directory che contiene i file Dockerfiles e Docker Compose.

5. Crea ed esegui contenitori

   Crea ed esegui i contenitori usando Docker Compose:

   ```
   docker-compose up --build
   ```

6. Fasi successive

   Ora puoi utilizzare l'agente Secrets Manager per recuperare segreti dal contenitore del tuo client. Per ulteriori informazioni, consulta the section called "Recupera i segreti con l'agente Secrets Manager".

Lambda

È possibile impacchettare l'agente Secrets Manager come estensione Lambda. Quindi puoi aggiungerlo alla tua funzione Lambda come livello e chiamare l'agente Secrets Manager dalla tua funzione Lambda per ottenere segreti.

Le seguenti istruzioni mostrano come ottenere un nome segreto MyTestutilizzando lo script `secrets-manager-agent-extension.sh` di esempio https://github.com/aws/aws-secretsmanager-agentper installare l'agente Secrets Manager come estensione Lambda.

Per creare un'estensione Lambda per l'agente Secrets Manager

1. Package dello strato agente

   Dalla radice del pacchetto di codice Secrets Manager Agent, esegui i seguenti comandi:

   ```
   AWS_ACCOUNT_ID=AWS_ACCOUNT_ID
   LAMBDA_ARN=LAMBDA_ARN

   # Build the release binary
   cargo build --release --target=x86_64-unknown-linux-gnu

   # Copy the release binary into the `bin` folder
   ```

```
mkdir -p ./bin
cp ./target/x86_64-unknown-linux-gnu/release/aws_secretsmanager_agent ./bin/
secrets-manager-agent

# Copy the `secrets-manager-agent-extension.sh` example script into the
 `extensions` folder.
mkdir -p ./extensions
cp aws_secretsmanager_agent/examples/example-lambda-extension/secrets-manager-
agent-extension.sh ./extensions

# Zip the extension shell script and the binary
zip secrets-manager-agent-extension.zip bin/* extensions/*

# Publish the layer version
LAYER_VERSION_ARN=$(aws lambda publish-layer-version \
    --layer-name secrets-manager-agent-extension \
    --zip-file "fileb://secrets-manager-agent-extension.zip" | jq -r
 '.LayerVersionArn')
```

2. Configurare il token SSRF

   La configurazione predefinita dell'agente imposterà automaticamente il token SSRF sul valore impostato nelle variabili preimpostate `AWS_SESSION_TOKEN` o di `AWS_CONTAINER_AUTHORIZATION_TOKEN` ambiente (quest'ultima variabile per le funzioni Lambda con abilitata). SnapStart In alternativa, puoi definire la variabile di `AWS_TOKEN` ambiente con un valore arbitrario per la tua funzione Lambda, poiché questa variabile ha la precedenza sulle altre due. Se si sceglie di utilizzare la variabile di `AWS_TOKEN` ambiente, è necessario impostare tale variabile di ambiente con una chiamata. `lambda:UpdateFunctionConfiguration`

3. Collega il livello alla funzione

   Collega la versione del layer alla tua funzione Lambda:

```
# Attach the layer version to the Lambda function
aws lambda update-function-configuration \
    --function-name $LAMBDA_ARN \
    --layers "$LAYER_VERSION_ARN"
```

4. Aggiornamento del codice della funzione

   Aggiorna la tua funzione Lambda per eseguire una query `http://localhost:2773/secretsmanager/get?secretId=MyTest` con il valore dell'`X-Aws-codes-Secrets-`

Tokenintestazione impostato sul valore del token SSRF proveniente da una delle variabili di ambiente sopra menzionate per recuperare il segreto. Assicurati di implementare la logica dei tentativi nel codice dell'applicazione per evitare ritardi nell'inizializzazione e nella registrazione dell'estensione Lambda.

5. Test della funzione

   Invocate la funzione Lambda per verificare che il segreto venga recuperato correttamente.

Recupera i segreti con l'agente Secrets Manager

Per recuperare un segreto, chiamate l'endpoint locale dell'agente Secrets Manager con il nome segreto o l'ARN come parametro di interrogazione. Per impostazione predefinita, l'agente Secrets Manager recupera la `AWSCURRENT` versione del segreto. Per recuperare una versione diversa, utilizzare il parametro VersionStage o VersionID.

> **Important**
>
> Per proteggere l'agente Secrets Manager, è necessario includere un'intestazione del token SSRF come parte di ogni richiesta:. `X-Aws-Parameters-Secrets-Token` L'agente Secrets Manager nega le richieste che non hanno questa intestazione o che hanno un token SSRF non valido. È possibile personalizzare il nome dell'intestazione SSRF in. the section called “Opzioni di configurazione”

Autorizzazioni richieste

L'agente Secrets Manager utilizza l' AWS SDK per Rust, che utilizza la catena di fornitori di AWS credenziali. L'identità di queste credenziali IAM determina le autorizzazioni di cui dispone l'agente Secrets Manager per recuperare i segreti.

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

Per ulteriori informazioni sulle autorizzazioni, consultare the section called “Riferimento per le autorizzazioni”.

> **Important**
>
> Dopo aver inserito il valore segreto nell'agente Secrets Manager, qualsiasi utente con accesso all'ambiente di calcolo e al token SSRF può accedere al segreto dalla cache di Secrets Manager Agent. Per ulteriori informazioni, consulta the section called "Considerazioni relative alla sicurezza".

Richieste di esempio

curl

Example Esempio: ottieni un segreto usando curl

Il seguente esempio curl mostra come ottenere un segreto dall'agente Secrets Manager. L'esempio si basa sulla presenza dell'SSRF in un file, che è dove viene memorizzato dallo script di installazione.

```
curl -v -H \\
    "X-Aws-Parameters-Secrets-Token: $(/var/run/awssmatoken)" \\
    'http://localhost:2773/secretsmanager/get?secretId=YOUR_SECRET_ID' \\
    echo
```

Python

Example Esempio: ottieni un segreto usando Python

Il seguente esempio in Python mostra come ottenere un segreto dall'agente Secrets Manager. L'esempio si basa sulla presenza dell'SSRF in un file, che è dove viene memorizzato dallo script di installazione.

```
import requests
import json

# Function that fetches the secret from Secrets Manager Agent for the provided
 secret id.
def get_secret():
    # Construct the URL for the GET request
    url = f"http://localhost:2773/secretsmanager/get?secretId=YOUR_SECRET_ID"
```

```
    # Get the SSRF token from the token file
    with open('/var/run/awssmatoken') as fp:
        token = fp.read()

    headers = {
        "X-Aws-Parameters-Secrets-Token": token.strip()
    }

    try:
        # Send the GET request with headers
        response = requests.get(url, headers=headers)

        # Check if the request was successful
        if response.status_code == 200:
            # Return the secret value
            return response.text
        else:
            # Handle error cases
            raise Exception(f"Status code {response.status_code} - {response.text}")

    except Exception as e:
        # Handle network errors
        raise Exception(f"Error: {e}")
```

Comprensione del parametro **`refreshNow`**

L'agente Secrets Manager utilizza una cache in memoria per archiviare i valori segreti, che aggiorna periodicamente. Per impostazione predefinita, questo aggiornamento si verifica quando si richiede un segreto dopo la scadenza del Time to Live (TTL), in genere ogni 300 secondi. Tuttavia, questo approccio a volte può portare a valori segreti obsoleti, specialmente se un segreto ruota prima della scadenza della voce della cache.

Per ovviare a questa limitazione, l'agente Secrets Manager supporta un parametro chiamato `refreshNow` nell'URL. È possibile utilizzare questo parametro per forzare l'aggiornamento immediato del valore di un segreto, aggirando la cache e assicurandosi di disporre della maggior parte delle up-to-date informazioni.

Comportamento predefinito (senza) **`refreshNow`**

- Utilizza i valori memorizzati nella cache fino alla scadenza del TTL
- Aggiorna i segreti solo dopo TTL (impostazione predefinita: 300 secondi)

- Può restituire valori obsoleti se i segreti ruotano prima della scadenza della cache

Comportamento con **`refreshNow=true`**

- Ignora completamente la cache
- Recupera l'ultimo valore segreto direttamente da Secrets Manager
- Aggiorna la cache con il nuovo valore e reimposta il TTL
- Ti assicura di ottenere sempre il valore segreto più recente

Aggiornamento forzato di un valore segreto

> **Important**
>
> Il valore predefinito di `refreshNow` è `false`. Se impostato su`true`, sovrascrive il TTL specificato nel file di configurazione di Secrets Manager Agent ed effettua una chiamata API a Secrets Manager.

curl

Example Esempio: aggiornamento forzato di un segreto utilizzando curl

Il seguente esempio curl mostra come forzare l'agente Secrets Manager ad aggiornare il segreto. L'esempio si basa sulla presenza dell'SSRF in un file, che è dove viene memorizzato dallo script di installazione.

```
curl -v -H \\
"X-Aws-Parameters-Secrets-Token: $(/var/run/awssmatoken)" \\
'http://localhost:2773/secretsmanager/get?secretId=YOUR_SECRET_ID&refreshNow=true' \
\
echo
```

Python

Example Esempio: aggiornamento forzato di un segreto usando Python

Il seguente esempio in Python mostra come ottenere un segreto dall'agente Secrets Manager. L'esempio si basa sulla presenza dell'SSRF in un file, che è dove viene memorizzato dallo script di installazione.

```
import requests
import json

# Function that fetches the secret from Secrets Manager Agent for the provided
 secret id.
def get_secret():
    # Construct the URL for the GET request
    url = f"http://localhost:2773/secretsmanager/get?
secretId=YOUR_SECRET_ID&refreshNow=true"

    # Get the SSRF token from the token file
    with open('/var/run/awssmatoken') as fp:
        token = fp.read()

    headers = {
        "X-Aws-Parameters-Secrets-Token": token.strip()
    }

    try:
        # Send the GET request with headers
        response = requests.get(url, headers=headers)

        # Check if the request was successful
        if response.status_code == 200:
            # Return the secret value
            return response.text
        else:
            # Handle error cases
            raise Exception(f"Status code {response.status_code} - {response.text}")

    except Exception as e:
        # Handle network errors
        raise Exception(f"Error: {e}")
```

Configurazione dell'agente Secrets Manager

Per modificare la configurazione dell'agente Secrets Manager, crea un file di configurazione TOML, quindi chiama. `./aws_secretsmanager_agent --config config.toml`

Opzioni di configurazione

`log_level`

Il livello di dettaglio riportato nei log per l'agente Secrets Manager: DEBUG, INFO, WARN, ERROR o NONE. L'impostazione predefinita è INFO.

`log_to_file`

Se accedere a un file o a stdout/stderr: o. `true` `false` Il valore predefinito è `true`.

`http_port`

La porta per il server HTTP locale, nell'intervallo da 1024 a 65535. L'impostazione predefinita è 2773.

`region`

La AWS regione da utilizzare per le richieste. Se non viene specificata alcuna regione, l'agente Secrets Manager determina la regione dall'SDK. Per ulteriori informazioni, consulta Specificare le credenziali e la regione predefinita nella Guida per sviluppatori AWS SDK for Rust.

`ttl_seconds`

Il TTL in secondi per gli elementi memorizzati nella cache, compreso tra 0 e 3600. L'impostazione predefinita è 300. 0 indica che non è presente alcuna memorizzazione nella cache.

`cache_size`

Il numero massimo di segreti che possono essere archiviati nella cache, compreso tra 1 e 1000. Il valore predefinito è 1000.

`ssrf_headers`

Un elenco di nomi di intestazione che l'agente Secrets Manager controlla per il token SSRF. L'impostazione predefinita è «X-Aws-Parameters-Secrets-Token,». X-Vault-Token

`ssrf_env_variables`

Un elenco di nomi di variabili di ambiente che l'agente Secrets Manager controlla in ordine sequenziale per il token SSRF. La variabile di ambiente può contenere il token o un riferimento al file del token come in:. `AWS_TOKEN=file:///var/run/awssmatoken` L'impostazione predefinita è "AWS_TOKEN, AWS_SESSION _TOKEN, AWS_CONTAINER _AUTHORIZATION_TOKEN».

`path_prefix`

Il prefisso URI utilizzato per determinare se la richiesta è una richiesta basata sul percorso. L'impostazione predefinita è «/v1/».

`max_conn`

Il numero massimo di connessioni dai client HTTP consentito dall'agente Secrets Manager, compreso tra 1 e 1000. Il valore predefinito è 800.

Funzionalità opzionali

L'agente Secrets Manager può essere creato con funzionalità opzionali passando il `--features` flag `acargo build`. Le funzionalità disponibili sono:

Caratteristiche di compilazione

`prefer-post-quantum`

Crea `X25519MLKEM768` l'algoritmo di scambio di chiavi con la massima priorità. Altrimenti, è disponibile ma non ha la massima priorità. `X25519MLKEM768`è un algoritmo ibrido di scambio di post-quantum-secure chiavi.

`fips`

Limita le suite di crittografia utilizzate dall'agente ai soli codici approvati dalla FIPS.

Registrazione dei log

Registrazione locale

L'agente Secrets Manager registra gli errori localmente nel file `logs/secrets_manager_agent.log` o su stdout/stderr a seconda della variabile di configurazione. `log_to_file` Quando l'applicazione chiama l'agente Secrets Manager per ottenere un segreto, tali chiamate vengono visualizzate nel registro locale. Non compaiono nei CloudTrail log.

Rotazione del registro

L'agente Secrets Manager crea un nuovo file di registro quando il file raggiunge i 10 MB e memorizza fino a cinque file di registro in totale.

AWS registrazione del servizio

Il registro non viene inviato a Secrets Manager CloudTrail, o CloudWatch. Le richieste di ottenere segreti dall'agente Secrets Manager non vengono visualizzate in questi registri. Quando l'agente Secrets Manager effettua una chiamata a Secrets Manager per ottenere un segreto, tale chiamata viene registrata CloudTrail con una stringa agente utente contenente`aws-secrets-manager-agent`.

È possibile configurare le opzioni di registrazione in. the section called "Opzioni di configurazione"

Considerazioni relative alla sicurezza

Dominio di fiducia

Per un'architettura ad agenti, il dominio di fiducia è il luogo in cui sono accessibili l'endpoint dell'agente e il token SSRF, che di solito è l'intero host. Il dominio di fiducia per l'agente Secrets Manager deve corrispondere al dominio in cui sono disponibili le credenziali di Secrets Manager per mantenere lo stesso livello di sicurezza. Ad esempio, su Amazon EC2 il dominio di fiducia per l'agente Secrets Manager sarebbe lo stesso del dominio delle credenziali quando si utilizzano i ruoli per Amazon EC2.

Important

Le applicazioni attente alla sicurezza che non utilizzano già una soluzione agente con le credenziali di Secrets Manager bloccate sull'applicazione dovrebbero prendere in considerazione l'utilizzo delle soluzioni specifiche del linguaggio o di memorizzazione AWS SDKs nella cache. Per ulteriori informazioni, consulta Get secrets.

Ottieni un valore segreto di Secrets Manager utilizzando l'SDK C++ AWS

Per le applicazioni C++, chiama l'SDK direttamente con o. GetSecretValueBatchGetSecretValue

I seguenti esempi di codice mostrano come recuperare un valore segreto di Gestione dei segreti.

Autorizzazioni richieste:`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();
}
```

Ottieni un valore segreto di Secrets Manager utilizzando l' JavaScript AWS SDK

Per JavaScript le applicazioni, chiama l'SDK direttamente con `getSecretValue`o. `batchGetSecretValue`

I seguenti esempi di codice mostrano come recuperare un valore segreto di Gestione dei segreti.

Autorizzazioni richieste:`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;
  }
};
```

Ottieni un valore segreto di Secrets Manager utilizzando l'SDK Kotlin AWS

Per le applicazioni Kotlin, chiama l'SDK direttamente con o. GetSecretValueBatchGetSecretValue

I seguenti esempi di codice mostrano come recuperare un valore segreto di Gestione dei segreti.

Autorizzazioni richieste:`secretsmanager:GetSecretValue`

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

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

Ottieni un valore segreto di Secrets Manager utilizzando l'SDK PHP AWS

Per le applicazioni PHP, effettua direttamente una chiamata all'SDK con `GetSecretValue` o `BatchGetSecretValue`.

I seguenti esempi di codice mostrano come recuperare un valore segreto di Gestione dei segreti.

Autorizzazioni richieste:`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
```

Ottieni un valore segreto di Secrets Manager usando Ruby SDK AWS

Per le applicazioni Ruby, effettua direttamente una chiamata all'SDK con `get_secret_value` o `batch_get_secret_value`.

I seguenti esempi di codice mostrano come recuperare un valore segreto di Gestione dei segreti.

Autorizzazioni richieste:`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
```

Ottieni un valore segreto usando il AWS CLI

Autorizzazioni richieste:`secretsmanager:GetSecretValue`

Example Recupero del valore del segreto crittografato di un segreto

L'esempio di `get-secret-value` seguente mostra come recuperare il valore corrente del segreto.

```
aws secretsmanager get-secret-value \
    --secret-id MyTestSecret
```

Example Recupero del valore del segreto precedente

L'esempio di `get-secret-value` seguente mostra come recuperare il valore precedente del segreto.

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

Ottieni un gruppo di segreti in un batch utilizzando il AWS CLI

Autorizzazioni richieste:

- `secretsmanager:BatchGetSecretValue`
- `secretsmanager:GetSecretValue`autorizzazione per ogni segreto che desideri recuperare.
- Se usi i filtri, devi avere anche `secretsmanager:ListSecrets`.

Per un esempio di policy delle autorizzazioni, consulta the section called "Esempio: autorizzazione a recuperare un gruppo di valori segreti in un batch".

> **Important**
>
> Se hai una policy VPCE che nega l'autorizzazione per recuperare un singolo segreto nel gruppo su cui stai agendo, `BatchGetSecretValue` non restituirà alcun valore segreto e restituirà un errore.

Example Recupera il valore segreto per un gruppo di segreti elencati per nome

L'esempio `batch-get-secret-value` seguente mostra come recuperare il valore corrente del segreto.

```
aws secretsmanager batch-get-secret-value \
          --secret-id-list MySecret1 MySecret2 MySecret3
```

Example Recupera il valore segreto per un gruppo di segreti selezionati dal filtro

L'esempio `batch-get-secret-value` seguente ottiene il valore segreto per i segreti che hanno un tag denominato "Test".

```
aws secretsmanager batch-get-secret-value \
          --filters Key="tag-key",Values="Test"
```

Ottieni un valore segreto usando la console AWS

Per recuperare un segreto (console)

1. Apri la console Secrets Manager all'indirizzo https://console.aws.amazon.com/secretsmanager/.
2. Nell'elenco di segreti, scegli il segreto che desideri recuperare.
3. Nella sezione Secret value (Valore segreto) scegli Retrieve secret value (Recupera valore segreto).

   Secrets Manager visualizza la versione corrente (`AWSCURRENT`) del segreto. Per visualizzare altre versioni del segreto, ad esempio `AWSPREVIOUS` versioni con etichetta personalizzata, usa the section called "AWS CLI".

Usa Gestione dei segreti AWS i segreti in AWS Batch

AWS Batch ti aiuta a eseguire carichi di lavoro di elaborazione in batch su. Cloud AWS Con AWS Batch, puoi inserire dati sensibili nei tuoi lavori archiviandoli in modo Gestione dei segreti AWS segreto e quindi facendo riferimento ad essi nella definizione del processo. Per ulteriori informazioni, consulta Specifica di dati sensibili con Secrets Manager.

Ottieni un Gestione dei segreti AWS segreto in una CloudFormation risorsa

Con CloudFormation, puoi recuperare un segreto da utilizzare in un'altra CloudFormation risorsa. Uno scenario comune consiste nel creare prima un segreto con una password generata da Secrets Manager e quindi recuperare il nome utente e la password dal segreto da utilizzare come credenziali per un nuovo database. Per informazioni sulla creazione di segreti con CloudFormation, consultaCloudFormation.

Per recuperare un segreto in un CloudFormation modello, si utilizza un riferimento dinamico. Quando create lo stack, il riferimento dinamico inserisce il valore segreto nella CloudFormation risorsa, quindi non è necessario codificare le informazioni segrete. Invece, riferisciti al segreto per nome o tramite l'ARN. È possibile utilizzare un riferimento dinamico per un segreto in qualsiasi proprietà della

risorsa. Non è possibile utilizzare un riferimento dinamico per un segreto nei metadati delle risorse, ad esempio `AWS::CloudFormation::Init`, perché ciò renderebbe visibile il valore del segreto nella console.

Un riferimento dinamico per un segreto ha il seguente modello:

```
{{resolve:secretsmanager:secret-id:SecretString:json-key:version-stage:version-id}}
```

secret-id

Il nome o l'ARN del segreto. Per accedere a un segreto nel tuo AWS account, puoi usare il nome segreto. Per accedere a un segreto in un altro AWS account, usa l'ARN del segreto.

json-key (Facoltativo)

Il nome della chiave della coppia chiave-valore di cui intendi recuperare il valore. Se non specifichi `ajson-key`, CloudFormation recupera l'intero testo segreto. Questo segmento può non includere il carattere di due punti (`:`).

version-stage (Facoltativo)

La versione del segreto da utilizzare. Secrets Manager utilizza le etichette temporanee per tenere traccia delle differenti versioni durante il processo di rotazione. Se utilizzi `version-stage`, non specificare `version-id`. Se non specifichi né `version-stage` né `version-id`, la versione di default sarà la `AWSCURRENT`. Questo segmento può non includere il carattere di due punti (`:`).

version-id (Facoltativo)

L'identificatore univoco della versione del segreto da utilizzare. Se specifichi `version-id`, non è necessario specificare anche `version-stage`. Se non specifichi né `version-stage` né `version-id`, la versione di default sarà la `AWSCURRENT`. Questo segmento può non includere il carattere di due punti (`:`).

Per ulteriori informazioni, consulta Utilizzo di riferimenti dinamici per specificare i segreti di Secrets Manager.

> **Note**
>
> Non create un riferimento dinamico utilizzando una barra rovesciata (`\`) come valore finale. CloudFormation non è in grado di risolvere tali riferimenti, il che causa un errore di risorse.

Usa Gestione dei segreti AWS i segreti nei GitHub lavori

Per utilizzare un segreto in un GitHub lavoro, puoi utilizzare un' GitHub azione per recuperare segreti da Gestione dei segreti AWS e aggiungerli come variabili di ambiente mascherate nel tuo GitHub flusso di lavoro. Per ulteriori informazioni sulle GitHub azioni, consulta Understanding GitHub Actions in the GitHub Docs.

Quando aggiungi un segreto al tuo GitHub ambiente, questo è disponibile per tutte le altre fasi del tuo GitHub lavoro. Segui le indicazioni contenute in Security Hardening for GitHub Actions per evitare che i segreti presenti nel tuo ambiente vengano utilizzati in modo improprio.

È possibile impostare l'intera stringa nel valore segreto come valore della variabile di ambiente oppure, se la stringa è JSON, è possibile analizzare il JSON per impostare variabili di ambiente individuali per ogni coppia chiave-valore JSON. Se il valore segreto è binario, l'operazione lo converte in una stringa.

Per visualizzare le variabili di ambiente create dai tuoi segreti, attiva la registrazione del debug. Per ulteriori informazioni, consulta Abilitazione della registrazione di debug nei Documenti. GitHub

Per utilizzare le variabili di ambiente create dai tuoi segreti, consulta Variabili di ambiente nei Documenti. GitHub

Prerequisiti

Per utilizzare questa azione, devi prima configurare AWS le credenziali e impostarle Regione AWS nel tuo GitHub ambiente utilizzando la `configure-aws-credentials` procedura. Segui le istruzioni riportate nella sezione Configurazione AWS delle credenziali Action for GitHub Actions to Assume il ruolo direttamente utilizzando il provider GitHub OIDC. Ciò consente di utilizzare credenziali di breve durata ed evitare di memorizzare chiavi di accesso aggiuntive al di fuori di Gestione dei segreti.

Il ruolo IAM assunto dall'operazione deve disporre delle seguenti autorizzazioni:

- `GetSecretValue` sui segreti che desideri recuperare.
- `ListSecrets` su tutti i segreti.
- (Facoltativo) `Decrypt` KMS key se i segreti sono crittografati con un. chiave gestita dal cliente

Per ulteriori informazioni, consulta the section called "Autenticazione e controllo degli accessi".

Utilizzo

Per utilizzare l'operazione, aggiungi un passaggio al flusso di lavoro che utilizza la seguente sintassi.

```
- 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
```

Parameters

`secret-ids`

ARN segreti, nomi e prefissi dei nomi.

Per impostare il nome della variabile di ambiente, inseriscilo prima dell'ID del segreto, seguito da una virgola. Ad esempio, `ENV_VAR_1, secretId` crea una variabile di ambiente denominata ENV_VAR_1 dal `secretId` del segreto. Il nome della variabile di ambiente può contenere lettere maiuscole, numeri e il carattere di sottolineatura.

Per utilizzare un prefisso, inserisci almeno tre caratteri seguiti da un asterisco. Ad esempio, `dev*` abbina tutti i segreti con un nome che inizia in dev. Possono essere recuperati fino a 100 segreti corrispondenti. Se imposti il nome della variabile e il prefisso trova corrispondenze con più segreti, l'operazione ha esito negativo.

`name-transformation`

Per impostazione predefinita, la fase crea ogni nome di variabile di ambiente dal nome del segreto, trasformato in modo da includere solo lettere maiuscole, numeri e caratteri di sottolineatura e in modo che non inizi con un numero. Per le lettere del nome, puoi configurare il passaggio in cui utilizzare lettere minuscole con `lowercase` o non modificare le lettere maiuscole con. `none` Il valore predefinito è `uppercase`.

`parse-json-secrets`

(Facoltativo) Per impostazione predefinita, l'operazione imposta il valore della variabile di ambiente sull'intera stringa JSON nel valore del segreto. Imposta `parse-json-secrets` per `true` creare variabili di ambiente per ogni coppia chiave-valore in JSON.

Ricorda che se il JSON utilizza chiavi con distinzione tra maiuscole e minuscole (come "nome" e "Nome"), l'operazione risconterà conflitti di nomi duplicati. In questo caso, imposta `parse-json-secrets` su `false` e analizza il valore segreto JSON separatamente.

Denominazione delle variabili di ambiente

Le variabili di ambiente create dall'azione hanno lo stesso nome dei segreti da cui provengono. Le variabili di ambiente hanno requisiti di denominazione più rigorosi rispetto ai segreti, quindi l'azione trasforma i nomi segreti per soddisfare tali requisiti. Ad esempio, l'operazione trasforma le lettere minuscole in lettere maiuscole. Se si analizza il codice JSON del segreto, il nome della variabile di ambiente include sia il nome segreto che il nome della chiave JSON, ad esempio. `MYSECRET_KEYNAME` È possibile configurare l'azione per non trasformare le lettere minuscole.

Se due variabili di ambiente finiscono con lo stesso nome, l'azione ha esito negativo. In questo caso, è necessario specificare i nomi che si desidera utilizzare per le variabili di ambiente come alias.

Esempi di casi in cui i nomi potrebbero essere in conflitto:

- Un segreto chiamato "MySecret" e un segreto chiamato «mysecret» diventerebbero entrambi variabili di ambiente denominate «MYSECRET».
- Un segreto denominato «SECRET_keyname» e un segreto analizzato in JSON denominato «Secret» con una chiave denominata «keyname» diventerebbero entrambi variabili di ambiente denominate «SECRET_KEYNAME».

È possibile impostare il nome della variabile di ambiente specificando un alias, come illustrato nell'esempio seguente, che crea una variabile denominata. `ENV_VAR_NAME`

```
secret-ids: |
  ENV_VAR_NAME, secretId2
```

Alias vuoti

- Se imposti `parse-json-secrets: true` e inserisci un alias vuoto, seguito da una virgola e quindi dall'ID segreto, l'azione assegna alla variabile di ambiente lo stesso nome delle chiavi JSON analizzate. I nomi delle variabili non includono il nome segreto.

  Se il segreto non contiene un codice JSON valido, l'azione crea una variabile di ambiente e la nomina con lo stesso nome del segreto.

- Se imposti `parse-json-secrets: false` e inserisci un alias vuoto, seguito da una virgola e dall'ID segreto, l'azione nomina le variabili di ambiente come se non avessi specificato un alias.

L'esempio seguente mostra un alias vuoto.

```
,secret2
```

Esempi

Example 1 Recupera segreti per nome e per ARN

L'esempio seguente crea variabili di ambiente per i segreti identificati dal nome e dall'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
```

Variabili di ambiente create:

```
EXAMPLESECRETNAME: secretValue1
TEST1: secretValue2
_0_TEST_SECRET: secretValue3
_PROD_EXAMPLE_SECRET: secretValue4
SECRET_ALIAS_1: secretValue5
SECRET_ALIAS_2: secretValue6
SECRET2: secretValue7
```

Example 2 Recupera tutti i segreti che iniziano con un prefisso

L'esempio seguente crea variabili di ambiente per tutti i segreti con nomi che iniziano con*beta*.

```
- name: Get Secret Names by Prefix
```

```
  uses: 2
  with:
    secret-ids: |
      beta*    # Retrieves all secrets that start with 'beta'
```

Variabili di ambiente create:

```
BETASECRETNAME: secretValue1
BETATEST: secretValue2
BETA_NEWSECRET: secretValue3
```

Example 3 Analizza il JSON nel segreto

L'esempio di seguito crea variabili di ambiente analizzando il JSON nel segreto.

```
- 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
```

Il segreto `test/secret` ha il seguente valore segreto.

```
{
  "api_user": "user",
  "api_key": "key",
  "config": {
    "active": "true"
  }
}
```

Il segreto `secret2` ha il seguente valore segreto.

```
{
  "myusername": "alejandro_rosalez",
  "mypassword": "EXAMPLE_PASSWORD"
}
```

Variabili di ambiente create:

```
TEST_SECRET_API_USER: "user"
TEST_SECRET_API_KEY: "key"
TEST_SECRET_CONFIG_ACTIVE: "true"
MYUSERNAME: "alejandro_rosalez"
MYPASSWORD: "EXAMPLE_PASSWORD"
```

Example 4 Usa lettere minuscole per i nomi delle variabili di ambiente

L'esempio seguente crea una variabile di ambiente con un nome minuscolo.

```
- name: Get secrets
  uses: aws-actions/aws-secretsmanager-get-secrets@v2
  with:
    secret-ids: exampleSecretName
    name-transformation: lowercase
```

Variabile di ambiente creata:

```
examplesecretname: secretValue
```

Usa Gestione dei segreti AWS in GitLab

Gestione dei segreti AWS si integra con GitLab. Puoi sfruttare i segreti di Secrets Manager per proteggere GitLab le tue credenziali in modo che non siano più codificate. GitLab Invece, GitLab Runner recupera questi segreti da Secrets Manager quando l'applicazione esegue un lavoro nelle pipeline GitLab CI/CD.

Per utilizzare questa integrazione, creerai un provider di identità OpenID Connect (OIDC) in IAM AWS Identity and Access Management e un ruolo IAM. Ciò consente a GitLab Runner di accedere al tuo segreto di Secrets Manager. Per ulteriori informazioni su GitLab CI/CD e OIDC, consulta la documentazione. GitLab

Considerazioni

Se utilizzi un' GitLab istanza non pubblica, non puoi utilizzare questa integrazione di Secrets Manager. Consulta invece la GitLab documentazione per le istanze non pubbliche.

Prerequisiti

Per integrare Secrets Manager con GitLab, completa i seguenti prerequisiti:

1. Crea un segreto Gestione dei segreti AWS

   Avrai bisogno di un segreto di Secrets Manager che verrà recuperato nel tuo GitLab lavoro ed elimina la necessità di codificare queste credenziali. Avrai bisogno dell'ID segreto di Secrets Manager per configurare la tua GitLab pipeline. Per ulteriori informazioni, consulta Crea un Gestione dei segreti AWS segreto.

2. Crea GitLab il tuo provider OIDC nella console IAM.

   In questo passaggio, creerai GitLab il tuo provider OIDC nella console IAM. Per ulteriori informazioni, consulta Create an Identity Provider OpenID Connect (OIDC) e documentazione. GitLab

   Quando crei il provider OIDC nella console IAM, utilizza le seguenti configurazioni:

   a. Imposta il sulla tua `provider URL` istanza. GitLab Ad esempio, **`gitlab.example.com`**.

   b. Imposta `audience` o `aud` su**`sts.amazonaws.com`**.

3. Crea un ruolo e una policy IAM

   Dovrai creare un ruolo e una policy IAM. Questo ruolo è assunto da GitLab with AWS Security Token Service (STS). Per ulteriori informazioni, consulta Creare un ruolo utilizzando politiche di fiducia personalizzate.

   a. Nella console IAM, utilizza le seguenti impostazioni durante la creazione del ruolo IAM:

      - Imposta `Trusted entity type` su **`Web identity`**.
      - Imposta `Group` su **`your GitLab group`**.
      - Imposta `Identity provider` lo stesso URL del provider (l'GitLab istanza) che hai utilizzato nel passaggio 2.
      - `Audience`Impostato sullo stesso pubblico utilizzato nel passaggio 2.

   b. Di seguito è riportato un esempio di politica di fiducia che consente di GitLab assumere ruoli. La tua politica di fiducia dovrebbe elencare Account AWS il tuo GitLab URL e il percorso del progetto.

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

```
    {
      "Effect": "Allow",
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:oidc-
provider/gitlab.example.com"
      },
      "Condition": {
        "StringEquals": {
          "gitlab.example.com:aud": [
            "sts.amazon.com"
          ]
        },
        "StringLike": {
          "gitlab.example.com:sub": [
            "project_path:mygroup/project-*:ref_type:branch-*:ref:main*"
          ]
        }
      }
    }
  ]
}
```

c. Dovrai anche creare una policy IAM per consentire GitLab l'accesso a Gestione dei segreti AWS. Puoi aggiungere questa politica alla tua politica di fiducia. Per ulteriori informazioni, consulta Creare politiche IAM.

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "secretsmanager:GetSecretValue",
      "Resource": "arn:aws:secretsmanager:us-
east-1:111122223333:secret:your-secret"
    }
  ]
}
```

Integrazione con Gestione dei segreti AWS GitLab

Dopo aver completato i prerequisiti, puoi configurare GitLab l'utilizzo di Secrets Manager per proteggere le tue credenziali.

Configurare la GitLab pipeline per utilizzare Secrets Manager

Dovrai aggiornare il file di configurazione GitLab CI/CD con le seguenti informazioni:

- Il pubblico del token impostato su STS.
- L'ID segreto di Secrets Manager.
- Il ruolo IAM che vuoi che GitLab Runner assuma durante l'esecuzione di lavori nella GitLab pipeline.
- Il Regione AWS luogo in cui è archiviato il segreto.

GitLab recupera il segreto da Secrets Manager e memorizza il valore in un file temporaneo. Il percorso di questo file è memorizzato in una CI/CD variabile, simile alle variabili CI/CD di tipo file.

Quanto segue è un frammento del file YAML per un file di configurazione CI/CD: GitLab

```
variables:
  AWS_REGION: us-east-1
  AWS_ROLE_ARN: 'arn:aws:iam::111122223333:role/gitlab-role'
job:
  id_tokens:
    AWS_ID_TOKEN:
      aud: 'sts.amazonaws.com'
  secrets:
    DATABASE_PASSWORD:
      aws_secrets_manager:
        secret_id: "arn:aws:secretsmanager:us-east-1:111122223333:secret:secret-name"
```

Per ulteriori informazioni, consulta la documentazione sull'integrazione di GitLab Secrets Manager.

Facoltativamente, puoi testare la tua configurazione OIDC in. GitLab Per ulteriori informazioni, GitLab consulta la documentazione per testare la configurazione OIDC.

Risoluzione dei problemi

Quanto segue può aiutarti a risolvere i problemi più comuni che potresti riscontrare durante l'integrazione di Secrets Manager con. GitLab

GitLab Problemi relativi alla pipeline

Se riscontri problemi relativi alla GitLab pipeline, accertati di quanto segue:

- Il file YAML è formattato correttamente. Per ulteriori informazioni, consulta la documentazione di GitLab.
- La tua GitLab pipeline sta assumendo il ruolo corretto, dispone delle autorizzazioni appropriate e dell'accesso al segreto corretto. Gestione dei segreti AWS

Risorse aggiuntive

Le seguenti risorse possono aiutarti a risolvere i problemi relativi a e: GitLab Gestione dei segreti AWS

- GitLab Risoluzione dei problemi OIDC
- GitLab Debug della pipeline CI/CD
- Risoluzione dei problemi

Usa Gestione dei segreti AWS i segreti in AWS IoT Greengrass

AWS IoT Greengrass è un software che estende le funzionalità cloud ai dispositivi locali. Consente ai dispositivi di raccogliere e analizzare i dati più vicini all'origine delle informazioni, reagire autonomamente a eventi locali e comunicare in modo sicuro tra di loro sulle reti locali.

AWS IoT Greengrass consente l'autenticazione con servizi e applicazioni da AWS IoT Greengrass dispositivi senza password, token o altri segreti codificati. Puoi utilizzarlo per archiviare e Gestione dei segreti AWS gestire in modo sicuro i tuoi segreti nel cloud. AWS IoT Greengrass estende Secrets Manager ai dispositivi AWS IoT Greengrass principali, in modo che i connettori e le funzioni Lambda possano utilizzare i segreti locali per interagire con servizi e applicazioni.

Per integrare un segreto in un AWS IoT Greengrass gruppo, si crea una risorsa di gruppo che fa riferimento al segreto di Secrets Manager. Questa risorsa segreta fa riferimento al segreto cloud

utilizzando l'ARN associato. Per informazioni su come creare, gestire e utilizzare risorse segrete, consulta Lavorare con le risorse segrete nella Guida per gli AWS IoT sviluppatori.

Per distribuire segreti nel AWS IoT Greengrass core, consulta Distribuire segreti nel AWS IoT Greengrass core.

Usa Gestione dei segreti AWS i segreti in Parameter Store

AWS Systems Manager Parameter Store fornisce uno storage sicuro e gerarchico per la gestione dei dati di configurazione e la gestione dei segreti. È possibile archiviare dati, ad esempio le password, le stringhe di database e i codici di licenza, come valori dei parametri. Tuttavia, Archivio parametri non fornisce servizi di rotazione automatica per i segreti archiviati. Al contrario, Archivio parametri consente di archiviare il segreto in Secrets Manager e quindi fare riferimento al segreto come a un parametro di Parameter Store.

Quando si configura l'archivio parametri con Secrets Manager, l'archivio parametri `secret-id` richiede una barra (/) prima della stringa del nome.

Per ulteriori informazioni, vedere Referencing Gestione dei segreti AWS Secrets from Parameters di Parameter Store nella Guida per l'AWS Systems Manager utente.

Ruota i segreti Gestione dei segreti AWS

La rotazione è il processo di aggiornamento periodico di un segreto. Quando si ruota un segreto, vengono aggiornati sia il segreto in Secrets Manager che le credenziali del database o del servizio. In Secrets Manager, è possibile impostare la rotazione automatica per i segreti. Esistono due forme di rotazione:

- Rotazione gestita— Per la maggior parte dei segreti gestiti, si utilizza la rotazione gestita, in cui il servizio configura e gestisce la rotazione per conto dell'utente. La rotazione gestita non utilizza una funzione Lambda.
- Segreti esterne gestite da Rotate Secrets Manager— Per i segreti detenuti dai partner di Secrets Manager, si utilizza la rotazione gestita dei segreti esterni per aggiornare il segreto sul sistema del partner. Ciò non richiede una funzione Lambda.
- the section called "Rotazione tramite funzione Lambda"— Per altri tipi di segreti, la rotazione di Secrets Manager utilizza una funzione Lambda per aggiornare il segreto e il database o il servizio.

Rotazione gestita per Gestione dei segreti AWS i segreti

Alcuni servizi offrono la rotazione gestita, in cui il servizio configura e gestisce la rotazione per tuo conto. Con la rotazione gestita, non si utilizza alcuna AWS Lambda funzione per aggiornare il segreto e le credenziali nel database.

I seguenti servizi offrono una rotazione gestita:

- Amazon Aurora offre una rotazione gestita per le credenziali degli utenti principali. Per ulteriori informazioni, consulta la sezione Gestione delle password con Amazon Aurora e Gestione dei segreti AWS nella Guida per l'utente di Amazon Aurora.
- Amazon ECS Service Connect offre una rotazione gestita per i certificati AWS Autorità di certificazione privata TLS. Per ulteriori informazioni, consulta TLS with Service Connect nella Amazon Elastic Container Service Developer Guide.
- Amazon RDS offre una rotazione gestita per le credenziali dell'utente principale. Per ulteriori informazioni, consulta la sezione Gestione delle password con Amazon RDS e Gestione dei segreti AWS nella Guida per l'utente di Amazon RDS.
- Amazon DocumentDB offre una rotazione gestita per le credenziali degli utenti principali. Per ulteriori informazioni, consulta Gestione delle password con Amazon DocumentDB e Gestione dei segreti AWS nella Guida per l'utente di Amazon DocumentDB.

- Amazon Redshift offre una rotazione gestita per le password degli amministratori. Per ulteriori informazioni, consulta Gestione delle password di amministratore Amazon Redshift utilizzando Gestione dei segreti AWS nella Guida alla gestione di Amazon Redshift.
- managed external secrets offre una rotazione gestita per i segreti detenuti dai partner di Secrets Manager. Per ulteriori informazioni, consulta Utilizzo di segreti esterni Gestione dei segreti AWS gestiti per gestire segreti di terze parti.

> **Tip**
>
> Per altri tipi di segreti, consulta la sezione the section called "Rotazione tramite funzione Lambda".

La rotazione per i segreti gestiti in genere viene completata entro un minuto. Durante la rotazione, le nuove connessioni che recuperano il segreto potrebbero ottenere la versione precedente delle credenziali. Nelle applicazioni, si consiglia di seguire la best practice di utilizzare un utente del database creato con i privilegi minimi richiesti per l'applicazione, piuttosto che utilizzare l'utente master. Per gli utenti dell'applicazione, ai fini della massima disponibilità, è possibile utilizzare la strategia di rotazione a utenti alternati.

Per i segreti detenuti dai partner di Secrets Manager,

Per modificare la pianificazione della rotazione gestita

1. Apri il segreto gestito nella console di Secrets Manager. Puoi seguire un link dal servizio di gestione o cercare il segreto nella console di Secrets Manager.
2. In Rotation schedule (Pianificazione della rotazione), inserisci la tua pianificazione seguendo il fuso orario UTC nel Schedule expression builder (Generatore di espressioni di pianificazione) o come Schedule expression (Espressione di pianificazione). Gestione dei segreti memorizza la tua pianificazione come un'espressione `rate()` o `cron()`. La finestra di rotazione inizia automaticamente a mezzanotte, a meno che non si specifichi un'ora di inizio. Puoi ruotare un segreto anche ogni quattro ore. Per ulteriori informazioni, consulta Pianificazioni di rotazione.
3. (Facoltativo) Per Window duration (Durata della finestra), scegli la lunghezza della finestra durante la quale vuoi che Secrets Manager ruoti il tuo segreto, ad esempio **3h** per una finestra di tre ore. La finestra non deve continuare nella finestra di rotazione successiva. Se non si specifica la durata della finestra, per un programma di rotazione espresso in ore, la finestra si chiude

automaticamente dopo un'ora. Per un programma di rotazione espresso in giorni, la finestra si chiude automaticamente alla fine della giornata.

4. Scegli Save (Salva).

Per modificare la pianificazione della rotazione gestita (AWS CLI)

- Chiama `rotate-secret`. L'esempio seguente mostra come ruotare il segreto tra le 16:00 e le 18:00 UTC del 1° e del 15° giorno del mese. Per ulteriori informazioni, consulta Pianificazioni di rotazione.

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

Segreti esterne gestite da Rotate Secrets Manager

Secrets Manager collabora con fornitori di software selezionati per offrire segreti esterni gestiti. Questa funzionalità aiuta i clienti a gestire il ciclo di vita segreto gestendo automaticamente le rotazioni. Con i segreti esterni gestiti, i clienti non devono più mantenere una logica di rotazione specifica per ogni segreto archiviato presso partner diversi. Questa operazione verrà gestita da Secrets Manager.

Per visualizzare l'elenco dei partner coinvolti con Secrets Manager, consulta Managed external secrets Partners.

Imposta la rotazione nella console

Per configurare la rotazione per un segreto esterno gestito esistente, creato specificando il tipo e il valore del segreto specificati dai rispettivi partner di integrazione, utilizza i seguenti passaggi:

1. Apri la console Secrets Manager.
2. Seleziona il tuo segreto esterno gestito dall'elenco.
3. Scegli la scheda Configurazione.
4. Nella sezione Configurazione della rotazione, scegli Modifica rotazione.
5. Attiva Automatic rotation (Rotazione automatica).

6. In Metadati di rotazione, aggiungi tutti i metadati specifici del partner necessari per la rotazione:

   Segui le linee guida fornite dal tuo partner di integrazione per gli altri metadati richiesti
7. In Autorizzazioni di servizio per la rotazione segreta, seleziona o crea un ruolo IAM per la rotazione:
   - Scegli Crea un nuovo ruolo per creare automaticamente un ruolo con le autorizzazioni necessarie
   - Oppure seleziona un ruolo esistente con le autorizzazioni appropriate per il tuo partner

   Per impostazione predefinita, le autorizzazioni sono limitate al singolo partner nella regione in cui viene creato il segreto
8. Imposta il tuo programma di rotazione (ad esempio, ruota automaticamente ogni 30 giorni).
9. Scegli Salva per applicare la configurazione di rotazione.

I due campi di metadati chiave configurati durante questo processo sono:

Campo	Description
ExternalSecretRotationMetadata	Metadati specifici per i partner necessari per la rotazione, ad esempio la versione API per Salesforce
ExternalSecretRotationRoleArn	L'ARN del ruolo IAM utilizzato per la rotazione, con autorizzazioni assegnate al partner di integrazione

Per ulteriori informazioni su questi campi, vedere Utilizzo dei segreti esterni gestiti da Secrets Manager per gestire i segreti di terze parti.

Configurare la rotazione utilizzando la CLI

Esegui il comando seguente per impostare la rotazione per un segreto di Salesforce. Questo comando specifica l'ID segreto, l'ARN del ruolo IAM per la rotazione, la pianificazione della rotazione e tutti i metadati specifici del partner richiesti per il processo di rotazione.

```
aws secretsmanager rotate-secret \
            --secret-id SampleSecret \
            --external-secret-rotation-role-arn arn:aws:iam::123412341234:role/xyz \
            --rotation-rules AutomaticallyAfterDays=1 \
```

```
            --external-secret-rotation-metadata
 '[{"Key":"apiVersion","Value":"v65.0"}]'
```

Rotazione tramite funzione Lambda

Per molti tipi di segreti, Secrets Manager utilizza una AWS Lambda funzione per aggiornare il segreto e il database o il servizio. Per ulteriori informazioni sui costi di utilizzo della funzione Lambda, consulta la sezione Prezzi.

Per alcuni Segreti gestiti da altri servizi, si utilizza la rotazione gestita. Per utilizzare Rotazione gestita, è necessario dapprima creare il segreto tramite il servizio di gestione.

Durante la rotazione, Secrets Manager registra gli eventi che indicano lo stato della rotazione. Per ulteriori informazioni, consulta the section called "Accedi con AWS CloudTrail ".

Per ruotare un segreto, Secrets Manager chiama una funzione Lambda in base alla pianificazione di rotazione impostata. Se aggiorni anche manualmente il valore segreto mentre è impostata la rotazione automatica, Secrets Manager la considera una rotazione valida quando calcola la data di rotazione successiva.

Durante la rotazione, Secrets Manager chiama la stessa funzione diverse volte, ogni volta con parametri diversi. Secrets Manager richiama la funzione con la struttura di parametri di richiesta JSON seguenti:

```
{
    "Step" : "request.type",
    "SecretId" : "string",
    "ClientRequestToken" : "string",
    "RotationToken" : "string"
}
```

Parametri:

- Fase: la fase di rotazione:`create_secret`, `set_secrettest_secret`, o. `finish_secret` Per ulteriori informazioni, consulta the section called "Quattro fasi in una funzione di rotazione".
- SecretId— L'ARN del segreto per ruotare.
- ClientRequestToken— Un identificatore univoco per la nuova versione del segreto. Questo valore aiuta a garantire l'idempotenza. Per ulteriori informazioni, consulta PutSecretValue: ClientRequestToken nel riferimento alle Gestione dei segreti AWS API.

- RotationToken— Un identificatore univoco che indica l'origine della richiesta. Richiesto per la rotazione segreta utilizzando un ruolo presunto o una rotazione tra account, in cui si ruota un segreto in un account utilizzando una funzione di rotazione Lambda in un altro account. In entrambi i casi, la funzione di rotazione assume un ruolo IAM per chiamare Secrets Manager, quindi Secrets Manager utilizza il token di rotazione per convalidare l'identità del ruolo IAM.

Se una qualsiasi fase di rotazione fallisce, Secrets Manager riprova l'intero processo di rotazione più volte.

Argomenti

Configura la rotazione automatica per i segreti di Amazon RDS, Amazon Aurora, Amazon Redshift o Amazon DocumentDB

Questo tutorial descrive come configurare i segreti the section called "Rotazione tramite funzione Lambda" del database. La rotazione è il processo di aggiornamento periodico di un segreto. Quando si ruota un segreto, vengono aggiornate le credenziali sia nel segreto che nel database. In Gestione dei segreti, puoi impostare la rotazione automatica per i segreti del tuo database.

Per impostare la rotazione utilizzando la console, prima è necessario scegliere una strategia di rotazione. Poi è possibile configurare il segreto per la rotazione, creando una funzione di rotazione Lambda se non è già presente. La console imposta anche le autorizzazioni per il ruolo di esecuzione della funzione Lambda. L'ultimo passaggio consiste nel garantire che la funzione di rotazione Lambda possa accedere sia a Gestione dei segreti che al database tramite la rete.

> **Warning**
>
> Per attivare la rotazione automatica, devi disporre dell'autorizzazione per creare un ruolo di esecuzione IAM per la funzione di rotazione Lambda e allegare una politica di autorizzazione. Sono necessarie entrambe le autorizzazioni `iam:CreateRole` e `iam:AttachRolePolicy`. La concessione di queste autorizzazioni consente a un'identità di concedersi qualsiasi autorizzazione.

Fasi:

- Fase 1: Scelta di una strategia di rotazione e (facoltativamente) creazione di un segreto del superutente
- Fase 2: Configurazione della rotazione e creazione di una funzione di rotazione
- Passaggio 3: (facoltativo) impostazione di condizioni di autorizzazione aggiuntive sulla funzione di rotazione
- Fase 4: Configurazione dell'accesso alla rete per la funzione di rotazione
- Fasi successive

Fase 1: Scelta di una strategia di rotazione e (facoltativamente) creazione di un segreto del superutente

Per informazioni sulle strategie offerte da Secrets Manager, vederethe section called "Strategie di rotazione delle funzioni Lambda".

Se scegli la strategia a utenti alternati, è necessario Crea segreti e memorizzare al suo interno le credenziali del superutente del database. Con le credenziali di superutente è necessario un segreto perché la rotazione clona il primo utente e la maggior parte degli utenti non dispone di tale autorizzazione. Tieni presente che Amazon RDS Proxy non supporta la strategia di utenti alternati.

Fase 2: Configurazione della rotazione e creazione di una funzione di rotazione

Per attivare la rotazione per un segreto Amazon RDS, Amazon DocumentDB o Amazon Redshift

1. Apri la console Secrets Manager all'indirizzo https://console.aws.amazon.com/secretsmanager/.
2. Nella pagina dell'elenco Secrets (Segreti) scegli il segreto.

3. Nella pagina Secret details (Dettagli del segreto), nella sezione Rotation configuration (Configurazione rotazione) scegli Edit rotation (Modifica rotazione).
4. Nella finestra di dialogo Edit rotation configuration (modifica configurazione rotazione), procedi come indicato di seguito:

   a. Attiva Automatic rotation (Rotazione automatica).

   b. In Rotation schedule (Pianificazione della rotazione), inserisci la tua pianificazione seguendo il fuso orario UTC nel Schedule expression builder (Generatore di espressioni di pianificazione) o come Schedule expression (Espressione di pianificazione). Gestione dei segreti memorizza la tua pianificazione come un'espressione `rate()` o `cron()`. La finestra di rotazione inizia automaticamente a mezzanotte, a meno che non si specifichi un'ora di inizio. Puoi ruotare un segreto anche ogni quattro ore. Per ulteriori informazioni, consulta Pianificazioni di rotazione.

   c. (Facoltativo) Per Window duration (Durata della finestra), scegli la lunghezza della finestra durante la quale vuoi che Secrets Manager ruoti il tuo segreto, ad esempio **3h** per una finestra di tre ore. La finestra non deve continuare nella finestra di rotazione successiva. Se non si specifica la durata della finestra, per un programma di rotazione espresso in ore, la finestra si chiude automaticamente dopo un'ora. Per un programma di rotazione espresso in giorni, la finestra si chiude automaticamente alla fine della giornata.

   d. (Facoltativo) Scegli Rotate immediately when the secret is stored (Ruota immediatamente quando viene memorizzato il segreto) per ruotare il segreto al salvataggio delle modifiche. Deselezionando la casella di controllo, la prima rotazione inizierà nella pianificazione impostata.

      Se la rotazione non avviene, ad esempio perché le fasi 3 e 4 non sono ancora state completate, Gestione dei segreti riprova il processo di rotazione più volte.

   e. In Rotation function (Funzione di rotazione), esegui una delle seguenti operazioni:

      - Scegli Crea una nuova funzione Lambda e immetti un nome per la nuova funzione. Gestione dei segreti aggiunge "`SecretsManager`" all'inizio del nome della funzione. Gestione dei segreti crea la funzione in base al modello appropriato e imposta le autorizzazioni necessarie per il ruolo di esecuzione di Lambda.
      - Scegli Use an existing Lambda function (Usa una funzione Lambda esistente) per riutilizzare una funzione di rotazione Lambda esistente per un altro segreto. Le funzioni di rotazione elencate in Recommended VPC configurations (Configurazioni VPC consigliate) dispongono dello stesso VPC e gruppo di sicurezza del database, e questo aiuta la funzione ad accedere al database.

f. Per Strategia di rotazione scegli la strategia Utente singolo o Utenti alternati. Per ulteriori informazioni, consulta the section called "Fase 1: Scelta di una strategia di rotazione e (facoltativamente) creazione di un segreto del superutente".

5. Scegli Save (Salva).

Passaggio 3: (facoltativo) impostazione di condizioni di autorizzazione aggiuntive sulla funzione di rotazione

Nella policy delle risorse per la funzione di rotazione, si consiglia di includere la chiave di contesto `aws:SourceAccount` per prevenire che Lambda venga usato come confused deputy. Per alcuni AWS servizi, per evitare il confuso scenario sostitutivo, si AWS consiglia di utilizzare sia i tasti di condizione `aws:SourceAccount`globale che i `aws:SourceArn`tasti di condizione globale. Tuttavia, se includi la condizione `aws:SourceArn` nella tua policy della funzione di rotazione, la funzione di rotazione può essere utilizzata solo per ruotare il segreto specificato da tale ARN. Ti consigliamo di includere solo la chiave di contesto `aws:SourceAccount` in modo da poter utilizzare la funzione di rotazione per più segreti.

Per aggiornare la policy delle risorse della funzione di rotazione

1. Nella console Gestione dei segreti, scegli il tuo segreto e quindi nella pagina dei dettagli, nella sezione Rotation configuration (Configurazione della rotazione), scegli la funzione di rotazione Lambda. Si apre la console Lambda.
2. Segui le istruzioni in Utilizzo di politiche basate sulle risorse per Lambda per aggiungere una condizione `aws:sourceAccount`.

```
"Condition": {
    "StringEquals": {
        "AWS:SourceAccount": "123456789012"
    }
},
```

Se il segreto è crittografato con una chiave KMS diversa da Chiave gestita da AWS `aws/secretsmanager`, Secrets Manager concede al ruolo di esecuzione Lambda l'autorizzazione a utilizzare la chiave. È possibile utilizzare il contesto di crittografia SecretARN per limitare l'uso della funzione di decrittografia, in modo che il ruolo della funzione di rotazione sia autorizzato ad accedere soltanto per decrittografare il segreto della cui rotazione è responsabile.

Aggiornamento del ruolo di esecuzione della funzione di rotazione

1. Dalla funzione di rotazione Lambda, scegli Configurazione, quindi in Ruolo di esecuzione scegli Nome del ruolo.
2. Segui le istruzioni riportate nella sezione Modifica di una policy sulle autorizzazioni dei ruoli per aggiungere una condizione `kms:EncryptionContext:SecretARN`.

```
"Condition": {
    "StringEquals": {
        "kms:EncryptionContext:SecretARN": "SecretARN"
    }
},
```

Fase 4: Configurazione dell'accesso alla rete per la funzione di rotazione

Per ulteriori informazioni, consulta the section called "Accesso alla rete per la funzione AWS Lambda di rotazione".

Fasi successive

Per informazioni, consulta the section called "Risoluzione dei problemi della rotazione".

Imposta la rotazione automatica per i segreti non relativi al database Gestione dei segreti AWS

Questo tutorial descrive come configurare i segreti non relativi the section called "Rotazione tramite funzione Lambda" al database. La rotazione è il processo di aggiornamento periodico di un segreto. Quando ruoti un segreto, aggiorni le credenziali sia nel segreto che nel database o nel servizio a cui si riferisce il segreto.

Per informazioni segrete sul database, consulta Rotazione automatica per i segreti del database (console).

> **Warning**
>
> Per attivare la rotazione automatica, devi disporre dell'autorizzazione per creare un ruolo di esecuzione IAM per la funzione di rotazione Lambda e allegare una politica di autorizzazione. Sono necessarie entrambe le autorizzazioni `iam:CreateRole` e `iam:AttachRolePolicy`.

La concessione di queste autorizzazioni consente a un'identità di concedersi qualsiasi autorizzazione.

Fasi:

- Fase 1: Creare una funzione di rotazione generica
- Fase 2: Scrittura del codice della funzione di rotazione
- Fase 3: Configurare il segreto per la rotazione
- Passaggio 4: consentire alla funzione di rotazione di accedere a Secrets Manager e al database o al servizio
- Passaggio 5: consentire a Secrets Manager di richiamare la funzione di rotazione
- Fase 6: Configurare l'accesso alla rete per la funzione di rotazione
- Fasi successive

Fase 1: Creare una funzione di rotazione generica

Per iniziare, crea una funzione di rotazione Lambda. Non conterrà il codice per ruotare il tuo segreto, quindi lo scriverai in un passaggio successivo. Per informazioni su come funziona una funzione di rotazione, consultathe section called "Funzioni di rotazione Lambda".

Nelle regioni supportate, è possibile utilizzare AWS Serverless Application Repository per creare la funzione da un modello. Per un elenco delle regioni supportate, consulta AWS Serverless Application Repository FAQs. In altre regioni, si crea la funzione da zero e si copia il codice del modello nella funzione.

Per creare una funzione di rotazione generica

1. Per determinare se AWS Serverless Application Repository è supportata nella tua regione, consulta gli AWS Serverless Application Repository endpoint e le quote nel Riferimento AWS generale.
2. Esegui una delle seguenti operazioni:
   - Se AWS Serverless Application Repository è supportato nella tua regione:
     a. Nella console Lambda, scegli Applicazioni, quindi scegli Crea applicazione.
     b. Nella pagina Crea applicazione, scegli la scheda Applicazione Serverless.

c. Nella casella di ricerca in Applicazioni pubbliche, inserisci**`SecretsManagerRotationTemplate`**.

d. Seleziona Mostra app che creano ruoli IAM personalizzati o politiche di risorse.

e. Seleziona in riquadro SecretsManagerRotationTemplate.

f. Nella pagina Rivedi, configura e distribuisci, nel riquadro Impostazioni dell'applicazione, compila i campi obbligatori.

- Per endpoint, inserisci l'endpoint per la tua regione, incluso. **`https://`** Per un elenco di endpoint, consulta the section called "Endpoint di Secrets Manager".
- Per inserire la funzione Lambda in un VPC, includi ID e. vpcSecurityGroup vpcSubnetIds

g. Seleziona Implementa.

- Se AWS Serverless Application Repository non è supportata nella tua regione:

a. Nella console Lambda, scegli Funzioni, quindi scegli Crea funzione.

b. Nella pagina Create function (Crea funzione), procedere come segue:

i. Scegli Author from scratch (Crea da zero).

ii. In Function name (Nome funzione), inserisci un nome per la funzione di rotazione.

iii. Per Runtime, scegli Python 3.10.

iv. Scegli Crea funzione.

Fase 2: Scrittura del codice della funzione di rotazione

In questo passaggio, scrivi il codice che aggiorna il segreto e il servizio o il database a cui è destinato il segreto. Per informazioni sul funzionamento di una funzione di rotazione, inclusi suggerimenti su come scrivere una funzione di rotazione personalizzata, vederethe section called "Funzioni di rotazione Lambda". È inoltre possibile utilizzarla Modelli di funzione di rotazione come riferimento.

Fase 3: Configurare il segreto per la rotazione

In questo passaggio, imposti un programma di rotazione per il tuo segreto e connetti la funzione di rotazione al segreto.

Per configurare la rotazione e creare una funzione di rotazione vuota

1. Apri la console Secrets Manager all'indirizzo https://console.aws.amazon.com/secretsmanager/.

2. Nella pagina dell'elenco Secrets (Segreti) scegli il segreto.
3. Nella pagina Secret details (Dettagli del segreto), nella sezione Rotation configuration (Configurazione rotazione) scegli Edit rotation (Modifica rotazione). Nella finestra di dialogo Edit rotation configuration (modifica configurazione rotazione), procedi come indicato di seguito:

   a. Attiva Automatic rotation (Rotazione automatica).

   b. In Rotation schedule (Pianificazione della rotazione), inserisci la tua pianificazione seguendo il fuso orario UTC nel Schedule expression builder (Generatore di espressioni di pianificazione) o come Schedule expression (Espressione di pianificazione). Gestione dei segreti memorizza la tua pianificazione come un'espressione `rate()` o `cron()`. La finestra di rotazione inizia automaticamente a mezzanotte, a meno che non si specifichi un'ora di inizio. Puoi ruotare un segreto anche ogni quattro ore. Per ulteriori informazioni, consulta Pianificazioni di rotazione.

   c. (Facoltativo) Per Window duration (Durata della finestra), scegli la lunghezza della finestra durante la quale vuoi che Secrets Manager ruoti il tuo segreto, ad esempio **3h** per una finestra di tre ore. La finestra non deve continuare nella finestra di rotazione successiva. Se non si specifica la durata della finestra, per un programma di rotazione espresso in ore, la finestra si chiude automaticamente dopo un'ora. Per un programma di rotazione espresso in giorni, la finestra si chiude automaticamente alla fine della giornata.

   d. (Facoltativo) Scegli Rotate immediately when the secret is stored (Ruota immediatamente quando viene memorizzato il segreto) per ruotare il segreto al salvataggio delle modifiche. Deselezionando la casella di controllo, la prima rotazione inizierà nella pianificazione impostata.

   e. In Funzione di rotazione, scegli la funzione Lambda che hai creato nel passaggio 1.

   f. Scegli Save (Salva).

Passaggio 4: consentire alla funzione di rotazione di accedere a Secrets Manager e al database o al servizio

La funzione di rotazione Lambda richiede l'autorizzazione per accedere al segreto in Gestione dei segreti e necessita dell'autorizzazione per accedere al database o al servizio. In questo passaggio, concedi queste autorizzazioni al ruolo di esecuzione di Lambda. Se il segreto è crittografato con una chiave KMS diversa da Chiave gestita da AWS `aws/secretsmanager`, allora è necessario concedere l'autorizzazione per utilizzare la chiave al ruolo di esecuzione Lambda. È possibile utilizzare il contesto di crittografia SecretARN per limitare l'uso della funzione di decrittografia, in

modo che il ruolo della funzione di rotazione sia autorizzato ad accedere soltanto per decrittografare il segreto della cui rotazione è responsabile. Per esempi di policy, consulta Autorizzazioni per la rotazione.

Per istruzioni, consulta Ruolo di esecuzione di Lambda nella Guida per gli sviluppatori di AWS Lambda .

Passaggio 5: consentire a Secrets Manager di richiamare la funzione di rotazione

Per consentire a Secrets Manager di richiamare la funzione di rotazione nella pianificazione di rotazione impostata, è necessario concedere l'`lambda:InvokeFunction`autorizzazione al responsabile del servizio Secrets Manager nella politica delle risorse della funzione Lambda.

Nella policy delle risorse per la funzione di rotazione, si consiglia di includere la chiave di contesto `aws:SourceAccount` per prevenire che Lambda venga usato come confused deputy. Per alcuni AWS servizi, per evitare il confuso scenario sostitutivo, si AWS consiglia di utilizzare sia i tasti di condizione `aws:SourceArn`sia i tasti di condizione `aws:SourceAccount`globali. Tuttavia, se includi la condizione `aws:SourceArn` nella tua policy della funzione di rotazione, la funzione di rotazione può essere utilizzata solo per ruotare il segreto specificato da tale ARN. Ti consigliamo di includere solo la chiave di contesto `aws:SourceAccount` in modo da poter utilizzare la funzione di rotazione per più segreti.

Per collegare una policy sulle risorse a una funzione Lambda, consulta Utilizzo delle policy basate su risorse per Lambda.

La seguente politica consente a Secrets Manager di richiamare una funzione Lambda.

JSON

```
{
    "Version":"2012-10-17",
    "Id": "default",
    "Statement": [
    {
        "Effect": "Allow",
        "Principal": {
            "Service": "secretsmanager.amazonaws.com"
            },
        "Action": "lambda:InvokeFunction",
        "Condition": {
```

```
            "StringEquals": {
                "AWS:SourceAccount": "123456789012"
            }
        },
        "Resource": "arn:aws:lambda:us-east-1:123456789012:function:function-
name"
    }
    ]
}
```

Fase 6: Configurare l'accesso alla rete per la funzione di rotazione

In questo passaggio, consenti alla funzione di rotazione di connettersi sia a Secrets Manager che al servizio o al database a cui è destinato il segreto. La funzione di rotazione deve avere accesso a entrambi per poter ruotare il segreto. Per informazioni, consulta the section called "Accesso alla rete per la funzione AWS Lambda di rotazione".

Fasi successive

Quando hai configurato la rotazione nel Passaggio 3, hai impostato una pianificazione per la rotazione del segreto. Se la rotazione fallisce quando è pianificata, Secrets Manager tenterà la rotazione più volte. È inoltre possibile avviare immediatamente una rotazione seguendo le istruzioni riportate inRotazione immediata di un segreto.

Se la rotazione fallisce, vediRisoluzione dei problemi della rotazione.

Imposta la rotazione automatica utilizzando il AWS CLI

Questo tutorial descrive come eseguire la configurazione the section called "Rotazione tramite funzione Lambda" utilizzando AWS CLI. Quando ruoti un segreto, aggiorni le credenziali sia nel segreto che nel database o nel servizio a cui si riferisce il segreto.

Puoi anche impostare la rotazione utilizzando la console. Per informazioni segrete sul database, consulta Rotazione automatica per i segreti del database (console). Per altri tipi di segreti, consulta la sezione Rotazione automatica per i segreti non relativi al database (console).

Per impostare la rotazione utilizzando AWS CLI, se si sta ruotando un database segreto, è necessario innanzitutto scegliere una strategia di rotazione. Se scegli la strategia a utenti alternati, è necessario archiviare un segreto separato con credenziali per un superutente del database. A questo punto, puoi scrivere il codice della funzione di rotazione. Gestione dei segreti fornisce modelli su cui

puoi basare la funzione. Quindi, crei una funzione Lambda con il tuo codice e imposti le autorizzazioni sia per la funzione Lambda che per il ruolo di esecuzione Lambda. Il passaggio successivo consiste nell'assicurarsi che la funzione Lambda possa accedere sia a Secrets Manager che al database o al servizio tramite la rete. Infine, puoi configurare il segreto per la rotazione.

Fasi:

- Prerequisito per i segreti del database: scegliere una strategia di rotazione
- Fase 1: scrivere il codice della funzione di rotazione
- Fase 2: Creare la funzione Lambda
- Passaggio 3: configurare l'accesso alla rete
- Fase 4: Configurare il segreto per la rotazione
- Fasi successive

Prerequisito per i segreti del database: scegliere una strategia di rotazione

Per informazioni sulle strategie offerte da Secrets Manager, vederethe section called "Strategie di rotazione delle funzioni Lambda".

Opzione 1: strategia per utente singolo

Se scegli la strategia per utente singolo, puoi continuare con la Fase 1.

Opzione 2: strategia per utenti alternati

Se scegli la strategia per utenti alternati, devi:

- Crea un segreto e memorizza le credenziali di superutente del database al suo interno. È necessario un segreto con credenziali di superutente perché la rotazione alternata degli utenti clona il primo utente e la maggior parte degli utenti non dispone di tale autorizzazione.
- Aggiungi l'ARN del segreto del superutente al segreto originale. Per ulteriori informazioni, consulta the section called "Struttura JSON di un segreto ".

Tieni presente che Amazon RDS Proxy non supporta la strategia di utenti alternati.

Fase 1: scrivere il codice della funzione di rotazione

Per ruotare un segreto, hai bisogno di una funzione di rotazione. Una funzione di rotazione è una funzione Lambda chiamata da Gestione dei segreti per ruotare il tuo segreto. Per ulteriori

informazioni, consulta the section called "Rotazione tramite funzione Lambda". In questo passaggio, scrivi il codice che aggiorna il segreto e il servizio o il database a cui è destinato il segreto.

Secrets Manager fornisce modelli per i segreti dei database Amazon RDS, Amazon Aurora, Amazon Redshift e Amazon DocumentDB. Modelli di funzione di rotazione

Per scrivere il codice della funzione di rotazione

1. Esegui una delle seguenti operazioni:
   - Controlla l'elenco dei modelli delle funzioni di rotazione. Se ce n'è uno che corrisponde al tuo servizio e alla tua strategia di rotazione, copia il codice.
   - Per altri tipi di segreti, scrivi la tua funzione di rotazione. Per istruzioni, consulta the section called "Funzioni di rotazione Lambda".
2. Salva il file in un file ZIP *my-function.zip* insieme a tutte le dipendenze richieste.

Fase 2: Creare la funzione Lambda

In questo passaggio, si crea la funzione Lambda utilizzando il file ZIP creato nel passaggio 1. È inoltre possibile impostare il ruolo di esecuzione Lambda, che è il ruolo che Lambda assume quando viene richiamata la funzione.

Creazione di una funzione di rotazione Lambda e di un ruolo di esecuzione

1. Crea una policy di attendibilità per il ruolo di esecuzione Lambda e salvalo come file JSON. Per esempi e maggiori informazioni, consulta the section called "Autorizzazioni per la rotazione". La policy deve:
   - Consentire al ruolo di chiamare le operazioni di Gestione dei segreti sul segreto.
   - Consenti al ruolo di chiamare il servizio a cui è destinato il segreto, ad esempio per creare una nuova password.
2. Crea il ruolo di esecuzione Lambda e applica la policy di fiducia creata nel passaggio precedente chiamando. `iam create-role`

```
aws iam create-role \
    --role-name rotation-lambda-role \
    --assume-role-policy-document file://trust-policy.json
```

3. Crea la funzione Lambda dal file ZIP chiamando `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. Imposta una policy delle risorse sulla funzione Lambda per consentire a Gestione dei segreti di richiamarla effettuando una chiamata `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
```

Passaggio 3: configurare l'accesso alla rete

Per ulteriori informazioni, consulta the section called "Accesso alla rete per la funzione AWS Lambda di rotazione".

Fase 4: Configurare il segreto per la rotazione

Per attivare la rotazione automatica del segreto, chiama `rotate-secret`. Puoi impostare una pianificazione di rotazione con un'espressione di pianificazione `cron()` o `rate()` e impostare una durata della finestra di rotazione. Per ulteriori informazioni, consulta the section called "Pianificazioni di rotazione".

```
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\"}"
```

Fasi successive

Per informazioni, consulta the section called "Risoluzione dei problemi della rotazione".

Strategie di rotazione delle funzioni Lambda

Infattithe section called "Rotazione tramite funzione Lambda", per quanto riguarda i segreti del database, Secrets Manager offre due strategie di rotazione.

Strategia di rotazione a utente singolo

Questa strategia aggiorna le credenziali per un utente in un unico segreto. Poiché gli utenti non possono modificare le proprie password, per le istanze Amazon RDS Db2 è necessario fornire le credenziali di amministratore in un segreto separato. Questa è la strategia di rotazione più semplice ed è adatta alla maggior parte dei casi d'uso. In particolare, ti consigliamo di utilizzare questa strategia per le credenziali per utenti occasionali (ad hoc) o interattivi.

Quando il segreto ruota, le connessioni aperte al database non vengono eliminate. Mentre si verifica la rotazione, tra la modifica della password nel database e l'aggiornamento del segreto corrispondente passa un breve periodo di tempo. Durante questo periodo di tempo, c'è un basso rischio che il database neghi le chiamate che utilizzano le credenziali ruotate. È possibile mitigare questo rischio con una strategia per nuovi tentativi appropriata. Dopo la rotazione, le nuove connessioni utilizzano le nuove credenziali.

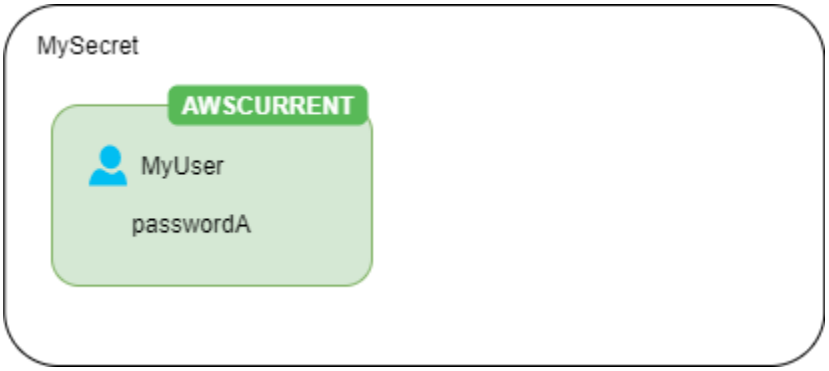
Strategia di rotazione a utenti alternati

Questa strategia aggiorna le credenziali per due utenti in un unico segreto. Viene creato il primo utente e, durante la prima rotazione, la funzione di rotazione lo clona creando il secondo. Ogni volta che il segreto ruota, la funzione di rotazione alterna quale password dell'utente viene aggiornata. Poiché la maggior parte degli utenti non dispone dell'autorizzazione per clonarsi, è necessario fornire le credenziali relative a un `superuser` in un altro segreto. Si consiglia di utilizzare la strategia di rotazione per singolo utente quando gli utenti clonati nel database non hanno le stesse autorizzazioni dell'utente originale e per le credenziali di utenti occasionali (ad hoc) o interattivi.

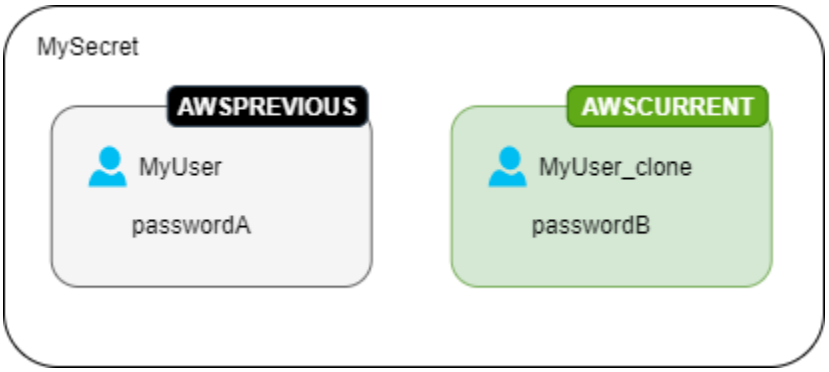
Questa strategia è adatta per i database con modelli di autorizzazione in cui un ruolo possiede le tabelle del database e un secondo ruolo ha l'autorizzazione per accedervi. È adatta anche all'uso in applicazioni che richiedono una disponibilità elevata. Se un'applicazione recupera il segreto durante la rotazione, ottiene comunque un set di credenziali valido. Dopo la rotazione, entrambe le credenziali `user` e `user_clone` sono valide. Ci sono anche meno possibilità che le applicazioni ottengano un rifiuto durante questo tipo di rotazione rispetto alla rotazione a utente singolo. Se il database è ospitato in una server farm dove la modifica della password richiede tempo per propagarsi a tutti i server, esiste il rischio che il database rifiuti le chiamate che utilizzano le nuove credenziali. È possibile mitigare questo rischio con una strategia per nuovi tentativi appropriata.

Secrets Manager crea l'utente clonato con le stesse autorizzazioni dell'utente originale. Se modifichi le autorizzazioni dell'utente originale dopo la creazione del clone, devi modificare anche le autorizzazioni dell'utente clonato.

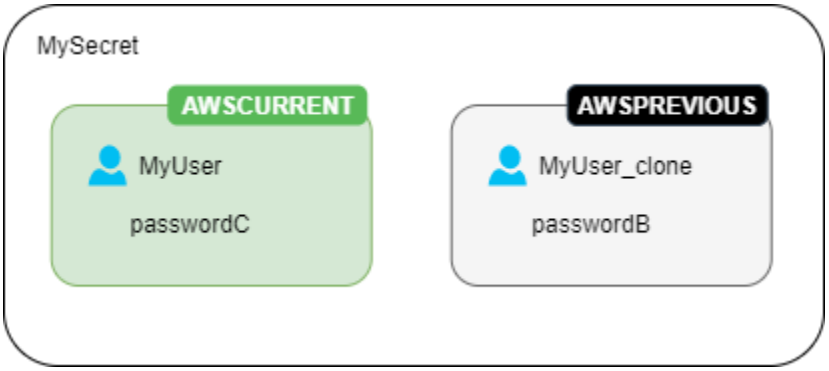
Ad esempio, se crei un segreto con le credenziali di un utente del database, il segreto contiene una versione con tali credenziali.



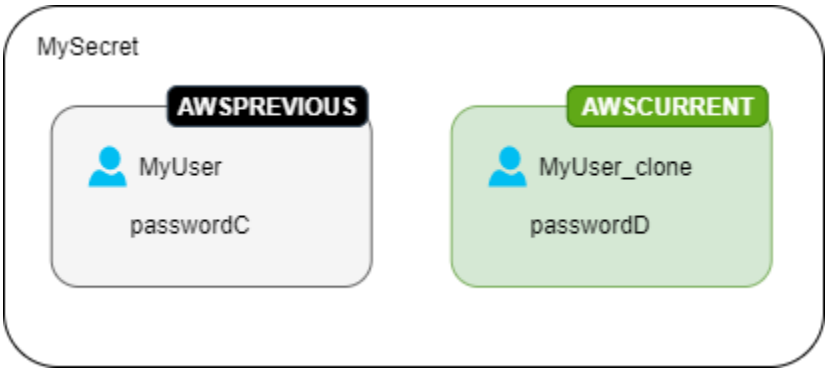
Prima rotazione: la funzione di rotazione crea un clone dell'utente con una password generata e tali credenziali diventano la versione segreta corrente.



Seconda rotazione: la funzione di rotazione aggiorna la password per l'utente originale.



Terza rotazione: la funzione di rotazione aggiorna la password per l'utente clonato.



Funzioni di rotazione Lambda

Inthe section called "Rotazione tramite funzione Lambda", una AWS Lambda funzione ruota il segreto. Gestione dei segreti AWS utilizza etichette temporanee per identificare le versioni segrete durante la rotazione.

Se Gestione dei segreti AWS non fornisce un modello di funzione di rotazione per il tipo segreto, puoi creare una funzione di rotazione personalizzata. Segui queste linee guida quando scrivi la tua funzione di rotazione:

Le migliori pratiche per le funzioni di rotazione personalizzate

- Usa il modello di rotazione generico come punto di partenza.
- Fai attenzione con le istruzioni di debug o di registrazione. Possono scrivere informazioni su Amazon CloudWatch Logs. Assicurati che i log non contengano informazioni sensibili.

  Per esempi di istruzioni di registro, consulta il codice the section called "Modelli di funzione di rotazione" sorgente.
- Per motivi di sicurezza, consente Gestione dei segreti AWS solo a una funzione di rotazione Lambda di ruotare direttamente il segreto. La funzione di rotazione non può chiamare un'altra funzione Lambda per ruotare il segreto.
- Per indicazioni sul debug, consulta Testing and debugging serverless applications.
- Se utilizzi binari e librerie esterne, ad esempio per connetterti a una risorsa, sei responsabile dell'applicazione di patch e del loro aggiornamento.
- Package della funzione di rotazione e di tutte le dipendenze in un file ZIP, ad esempio*my-function.zip*.

> **Warning**
>
> L'impostazione del parametro di concorrenza assegnato su un valore inferiore a 10 può causare una limitazione a causa di thread di esecuzione insufficienti per la funzione Lambda. Per ulteriori informazioni, consulta Understanding Reserved Concurrency e Provisioned Concurrency nella Developer Guide. AWS Lambda AWS Lambda

Quattro fasi in una funzione di rotazione

Argomenti

- createSecret: crea una nuova versione del segreto
- setSecret: modifica le credenziali nel database o nel servizio
- testSecret: prova la nuova versione segreta
- finishSecret: Termina la rotazione

`createSecret`: crea una nuova versione del segreto

Il metodo verifica `createSecret` innanzitutto se esiste un segreto chiamando `get_secret_value`con il pass-in`ClientRequestToken`. Se non c'è alcun segreto, crea un nuovo segreto con `create_secret`e il token come. `VersionId` Quindi genera un nuovo valore segreto con `get_random_password`. Successivamente chiama `put_secret_value`per memorizzarlo con l'etichetta `AWSPENDING` di staging. La memorizzazione del nuovo valore del segreto in `AWSPENDING` aiuta a garantire l'idempotenza. Se per un motivo qualsiasi la rotazione non viene eseguita, puoi fare riferimento a quel valore del segreto nelle chiamate successive. Consulta Come posso rendere idempotente la mia funzione Lambda.

Suggerimenti per scrivere una funzione di rotazione personalizzata

- Assicurati che il nuovo valore segreto includa solo caratteri validi per il database o il servizio. Escludi i caratteri utilizzando il parametro `ExcludeCharacters`.
- Mentre testate la vostra funzione, utilizzate le fasi AWS CLI per vedere la versione: chiamate `describe-secret`e guardate`VersionIdsToStages`.
- Per Amazon RDS MySQL, nella rotazione alternata degli utenti, Secrets Manager crea un utente clonato con un nome non più lungo di 16 caratteri. Puoi modificare la funzione di rotazione per consentire nomi utente più lunghi. La versione 5.7 e successive di MySQL supportano nomi utente

fino a 32 caratteri, tuttavia Secrets Manager aggiunge "_clone" (sei caratteri) alla fine del nome utente, quindi è necessario mantenere il nome utente a un massimo di 26 caratteri.

setSecret: modifica le credenziali nel database o nel servizio

Il metodo `setSecret` modifica la credenziale nel database o nel servizio in modo che corrisponda al nuovo valore segreto nella `AWSPENDING` versione del segreto.

Suggerimenti per scrivere una funzione di rotazione personalizzata

- Se passate istruzioni a un servizio che interpreta le istruzioni, ad esempio un database, utilizzate la parametrizzazione delle query. Per ulteriori informazioni, vedere Query Parameterization Cheat Sheet sul sito Web OWASP.
- La funzione di rotazione è un "privileged deputy" che ha l'autorizzazione ad accedere e modificare le credenziali del cliente sia nel segreto di Gestione dei segreti che nella risorsa di destinazione. Per evitare un potenziale attacco confused deputy, devi assicurarti che un utente malintenzionato non possa utilizzare la funzione per accedere ad altre risorse. Prima di aggiornare le credenziali:
  - Verifica che la credenziale nella versione `AWSCURRENT` del segreto sia valida. Se la credenziale `AWSCURRENT` non è valida, abbandona il tentativo di rotazione.
  - Verifica che i valori dei segreti `AWSCURRENT` e `AWSPENDING` si riferiscano alla stessa risorsa. Per il nome utente e la password, verifica che i nomi utente `AWSCURRENT` e `AWSPENDING` siano uguali.
  - Verifica che la risorsa del servizio di destinazione sia la stessa. Per un database, verifica che i nomi degli host `AWSCURRENT` e `AWSPENDING` siano uguali.
- In rari casi, potresti voler personalizzare una funzione di rotazione esistente per un database. Ad esempio, con la rotazione alternata degli utenti, Secrets Manager crea l'utente clonato copiando i parametri di configurazione di runtime del primo utente. Se desideri includere più attributi o modificare quelli concessi all'utente clonato, devi aggiornare il codice nella funzione `set_secret`.

testSecret: prova la nuova versione segreta

Successivamente, la funzione di rotazione Lambda esegue il test della versione `AWSPENDING` del segreto utilizzando questa versione per accedere al database o al servizio. Le funzioni di rotazione basate su Modelli di funzione di rotazione verificano il nuovo segreto utilizzando un accesso in lettura.

finishSecret: Termina la rotazione

Infine, la funzione di rotazione Lambda sposta l'etichetta `AWSCURRENT` dalla precedente versione segreta a questa versione, che rimuove anche l'etichetta `AWSPENDING` nella stessa chiamata API. Secrets Manager aggiunge l'etichetta di gestione temporanea `AWSPREVIOUS` alla versione precedente, in modo da conservare l'ultima versione nota del segreto.

Il metodo finish_secret consente `update_secret_version_stage`di spostare l'etichetta di staging `AWSCURRENT` dalla versione segreta precedente alla nuova versione segreta. Gestione dei segreti aggiunge automaticamente l'etichetta di gestione temporanea `AWSPREVIOUS` alla versione precedente, in modo da conservare l'ultima versione nota del segreto.

Suggerimenti per scrivere una funzione di rotazione personalizzata

- Non rimuovere `AWSPENDING` prima di questo punto e non rimuoverlo utilizzando una chiamata API separata, poiché ciò può indicare a Secrets Manager che la rotazione non è stata completata correttamente. Secrets Manager aggiunge l'etichetta di gestione temporanea `AWSPREVIOUS` alla versione precedente, in modo da conservare l'ultima versione nota del segreto.

Quando la rotazione ha esito positivo, l'etichetta di gestione temporanea `AWSPENDING` potrebbe essere collegata alla stessa versione come la versione `AWSCURRENT`, oppure potrebbe non essere collegata a nessuna versione. Se l'etichetta di gestione temporanea `AWSPENDING` è presente ma non è collegata alla stessa versione come `AWSCURRENT`, qualsiasi successiva chiamata di rotazione presuppone che una precedente richiesta di rotazione sia ancora in corso e viene segnalato un errore. Quando la rotazione non ha esito positivo, l'etichetta di gestione temporanea `AWSPENDING` potrebbe essere collegata a una versione di un segreto vuota. Per ulteriori informazioni, consulta Risoluzione dei problemi della rotazione.

Gestione dei segreti AWS modelli di funzioni di rotazione

Gestione dei segreti AWS fornisce una serie di modelli di funzioni di rotazione che aiutano ad automatizzare la gestione sicura delle credenziali per vari sistemi e servizi di database. I modelli sono funzioni ready-to-use Lambda che implementano le migliori pratiche per la rotazione delle credenziali, aiutandoti a mantenere il tuo livello di sicurezza senza interventi manuali.

I modelli supportano due strategie di rotazione principali:

- Rotazione per utente singolo che aggiorna le credenziali per un singolo utente.

- Rotazione tra utenti alternati che mantiene due utenti separati per eliminare i tempi di inattività durante la modifica delle credenziali.

Secrets Manager fornisce anche un modello generico che funge da punto di partenza per qualsiasi tipo di segreto.

Per utilizzare il modello, consulta:

- Rotazione automatica per i segreti del database (console)
- Rotazione automatica per i segreti non relativi al database (console)

Per scrivere una funzione di rotazione personalizzata, consultate Scrivere una funzione di rotazione.

Modelli

- Amazon RDS e Amazon Aurora
  - Amazon RDS Db2 per utente singolo
  - Amazon RDS Db2 utenti alternati
  - Utente singolo di Amazon RDS MariaDB
  - Utenti alternativi di Amazon RDS MariaDB
  - Utente singolo di Amazon RDS e Amazon Aurora MySQL
  - Utenti alternati di Amazon RDS e Amazon Aurora MySQL
  - Utente singolo per Amazon RDS Oracle
  - Utenti alternati Amazon RDS Oracle
  - Utente singolo di Amazon RDS e Amazon Aurora PostgreSQL
  - Utenti alternati di Amazon RDS e Amazon Aurora PostgreSQL
  - Amazon RDS Microsoft per utente SQLServer singolo
  - Amazon RDS Microsoft utenti SQLServer alternati
- Amazon DocumentDB (compatibile con MongoDB)
  - Utente singolo Amazon DocumentDB
  - Utenti alternativi Amazon DocumentDB
- Amazon Redshift
  - Amazon Redshift utente singolo
  - Utenti alternati Amazon Redshift

- Amazon Timestream per InfluxDB
  - Amazon Timestream per utente singolo InfluxDB
  - Amazon Timestream per utenti alternati di InfluxDB
- Amazon ElastiCache
- Active Directory
  - Credenziali Active Directory
  - Tastiera Active Directory
- Other type of secret (Altro tipo di segreto)

Amazon RDS e Amazon Aurora

Amazon RDS Db2 per utente singolo

- Nome del modello: SecretsManager RDSDb2 RotationSingleUser
- Strategia di rotazione: Strategia di rotazione a utente singolo.
- Struttura del **`SecretString`**: the section called "Credenziali Amazon RDS e Aurora".
- Codice sorgente: https://github.com/aws-samples/ aws-secrets-manager-rotation- lambdas/tree/master/SecretsManagerRDSDb2RotationSingleUser/lambda _function.py
- Dipendenza: python-ibmdb

Amazon RDS Db2 utenti alternati

- Nome del modello: SecretsManager RDSDb2 RotationMultiUser
- Strategia di rotazione: the section called "Utenti alternati".
- Struttura del **`SecretString`**: the section called "Credenziali Amazon RDS e Aurora".
- Codice sorgente: https://github.com/aws-samples/ aws-secrets-manager-rotation- lambdas/tree/master/SecretsManagerRDSDb2RotationMultiUser/lambda _function.py
- Dipendenza: python-ibmdb

Utente singolo di Amazon RDS MariaDB

- Nome del modello: SecretsManager RDSMaria DBRotation SingleUser
- Strategia di rotazione: Strategia di rotazione a utente singolo.

- Struttura del **`SecretString`**: the section called "Credenziali Amazon RDS e Aurora".
- Codice sorgente: https://github.com/aws-samples/ aws-secrets-manager-rotation- lambdas/tree/master/SecretsManagerRDSMariaDBRotationSingleUser/lambda _function.py
- Dipendenza: PyMy SQL 1.0.2. Se si utilizza la password sha256 per l'autenticazione, PyMy SQL [rsa]. Per informazioni sull'utilizzo di pacchetti con codice compilato in un runtime Lambda, vedi Come posso aggiungere pacchetti Python con binari compilati al mio pacchetto di distribuzione e rendere il pacchetto compatibile con Lambda? nel Knowledge Center.AWS

Utenti alternativi di Amazon RDS MariaDB

- Nome del modello: SecretsManager RDSMaria DBRotation MultiUser
- Strategia di rotazione: the section called "Utenti alternati".
- Struttura del **`SecretString`**: the section called "Credenziali Amazon RDS e Aurora".
- Codice sorgente: https://github.com/aws-samples/ aws-secrets-manager-rotation- lambdas/tree/master/SecretsManagerRDSMariaDBRotationMultiUser/lambda _function.py
- Dipendenza: PyMy SQL 1.0.2. Se si utilizza la password sha256 per l'autenticazione, PyMy SQL [rsa]. Per informazioni sull'utilizzo di pacchetti con codice compilato in un runtime Lambda, vedi Come posso aggiungere pacchetti Python con binari compilati al mio pacchetto di distribuzione e rendere il pacchetto compatibile con Lambda? nel Knowledge Center.AWS

Utente singolo di Amazon RDS e Amazon Aurora MySQL

- Nome del modello: SecretsManager RDSMy SQLRotation SingleUser
- Strategia di rotazione: the section called "Utente singolo".
- Struttura del **`SecretString`** prevista: the section called "Credenziali Amazon RDS e Aurora".
- Codice sorgente: https://github.com/aws-samples/ aws-secrets-manager-rotation- lambdas/tree/master/SecretsManagerRDSMySQLRotationSingleUser/lambda _function.py
- Dipendenza: PyMy SQL 1.0.2. Se si utilizza la password sha256 per l'autenticazione, PyMy SQL [rsa]. Per informazioni sull'utilizzo di pacchetti con codice compilato in un runtime Lambda, vedi Come posso aggiungere pacchetti Python con binari compilati al mio pacchetto di distribuzione e rendere il pacchetto compatibile con Lambda? nel Knowledge Center.AWS

Utenti alternati di Amazon RDS e Amazon Aurora MySQL

- Nome del modello: SecretsManager RDSMy SQLRotation MultiUser
- Strategia di rotazione: the section called "Utenti alternati".
- Struttura del **`SecretString`** prevista: the section called "Credenziali Amazon RDS e Aurora".
- Codice sorgente: https://github.com/aws-samples/ aws-secrets-manager-rotation- lambdas/tree/master/SecretsManagerRDSMySQLRotationMultiUser/lambda _function.py
- Dipendenza: PyMy SQL 1.0.2. Se si utilizza la password sha256 per l'autenticazione, PyMy SQL [rsa]. Per informazioni sull'utilizzo di pacchetti con codice compilato in un runtime Lambda, vedi Come posso aggiungere pacchetti Python con binari compilati al mio pacchetto di distribuzione e rendere il pacchetto compatibile con Lambda? nel Knowledge Center.AWS

Utente singolo per Amazon RDS Oracle

- Nome del modello: SecretsManager RDSOracle RotationSingleUser
- Strategia di rotazione: the section called "Utente singolo".
- Struttura del **`SecretString`** prevista: the section called "Credenziali Amazon RDS e Aurora".
- Codice sorgente: https://github.com/aws-samples/ aws-secrets-manager-rotation- lambdas/tree/master/SecretsManagerRDSOracleRotationSingleUser/lambda _function.py
- Dipendenza: python-oracledb 2.4.1

Utenti alternati Amazon RDS Oracle

- Nome del modello: SecretsManager RDSOracle RotationMultiUser
- Strategia di rotazione: the section called "Utenti alternati".
- Struttura del **`SecretString`** prevista: the section called "Credenziali Amazon RDS e Aurora".
- Codice sorgente: https://github.com/aws-samples/ aws-secrets-manager-rotation- lambdas/tree/master/SecretsManagerRDSOracleRotationMultiUser/lambda _function.py
- Dipendenza: python-oracledb 2.4.1

Utente singolo di Amazon RDS e Amazon Aurora PostgreSQL

- Nome del modello: SecretsManager RDSPostgre SQLRotation SingleUser
- Strategia di rotazione: Strategia di rotazione a utente singolo.

- Struttura del **`SecretString`** prevista: the section called "Credenziali Amazon RDS e Aurora".
- Codice sorgente: https://github.com/aws-samples/ aws-secrets-manager-rotation- lambdas/tree/master/SecretsManagerRDSPostgreSQLRotationSingleUser/lambda _function.py
- Dipendenza: PyGre SQL 5.2.5

Utenti alternati di Amazon RDS e Amazon Aurora PostgreSQL

- Nome del modello: SecretsManager RDSPostgre SQLRotation MultiUser
- Strategia di rotazione: the section called "Utenti alternati".
- Struttura del **`SecretString`** prevista: the section called "Credenziali Amazon RDS e Aurora".
- Codice sorgente: https://github.com/aws-samples/ aws-secrets-manager-rotation- lambdas/tree/master/SecretsManagerRDSPostgreSQLRotationMultiUser/lambda _function.py
- Dipendenza: PyGre SQL 5.2.5

Amazon RDS Microsoft per utente SQLServer singolo

- Nome del modello: SecretsManager RDSSQLServer RotationSingleUser
- Strategia di rotazione: the section called "Utente singolo".
- Struttura del **`SecretString`** prevista: the section called "Credenziali Amazon RDS e Aurora".
- Codice sorgente: https://github.com/aws-samples/ aws-secrets-manager-rotation- lambdas/tree/master/SecretsManagerRDSSQLServerRotationSingleUser/lambda _function.py
- Dipendenza: Pymssql 2.2.2

Amazon RDS Microsoft utenti SQLServer alternati

- Nome del modello: SecretsManager RDSSQLServer RotationMultiUser
- Strategia di rotazione: the section called "Utenti alternati".
- Struttura del **`SecretString`** prevista: the section called "Credenziali Amazon RDS e Aurora".
- Codice sorgente: https://github.com/aws-samples/ aws-secrets-manager-rotation- lambdas/tree/master/SecretsManagerRDSSQLServerRotationMultiUser/lambda _function.py
- Dipendenza: Pymssql 2.2.2

Amazon DocumentDB (compatibile con MongoDB)

Utente singolo Amazon DocumentDB

- Nome del modello: SecretsManagerMongo DBRotation SingleUser
- Strategia di rotazione: the section called "Utente singolo".
- Struttura del **`SecretString`** prevista: the section called "Credenziali Amazon DocumentDB".
- Codice sorgente: https://github.com/aws-samples/ aws-secrets-manager-rotation- lambdas/tree/master/SecretsManagerMongoDBRotationSingleUser/lambda _function.py
- Dipendenza: 4.2.0 PyMongo

Utenti alternativi Amazon DocumentDB

- Nome del modello: SecretsManagerMongo DBRotation MultiUser
- Strategia di rotazione: the section called "Utenti alternati".
- Struttura del **`SecretString`** prevista: the section called "Credenziali Amazon DocumentDB".
- Codice sorgente: https://github.com/aws-samples/ aws-secrets-manager-rotation- lambdas/tree/master/SecretsManagerMongoDBRotationMultiUser/lambda _function.py
- Dipendenza: 4.2.0 PyMongo

Amazon Redshift

Amazon Redshift utente singolo

- Nome del modello: SecretsManagerRedshiftRotationSingleUser
- Strategia di rotazione: the section called "Utente singolo".
- Struttura del **`SecretString`** prevista: the section called "Credenziali Amazon Redshift".
- Codice sorgente: https://github.com/aws-samples/ aws-secrets-manager-rotation- lambdas/tree/master/SecretsManagerRedshiftRotationSingleUser/lambda _function.py
- Dipendenza: PyGre SQL 5.2.5

Utenti alternati Amazon Redshift

- Nome del modello: SecretsManagerRedshiftRotationMultiUser

- Strategia di rotazione: the section called "Utenti alternati".
- Struttura del **`SecretString`** prevista: the section called "Credenziali Amazon Redshift".
- Codice sorgente: https://github.com/aws-samples/ aws-secrets-manager-rotation- lambdas/tree/master/SecretsManagerRedshiftRotationMultiUser/lambda _function.py
- Dipendenza: PyGre SQL 5.2.5

Amazon Timestream per InfluxDB

Per utilizzare questi modelli, consulta l'articolo Come Amazon Timestream for InfluxDB utilizza i segreti nella Amazon Timestream Developer Guide.

Amazon Timestream per utente singolo InfluxDB

- Nome del modello: Influx SecretsManager DBRotation SingleUser
- Struttura del **`SecretString`** prevista: the section called "Amazon Timestream per la struttura segreta di InfluxDB".
- Codice sorgente: https://github.com/aws-samples/ aws-secrets-manager-rotation- _function.py lambdas/tree/master/SecretsManagerInfluxDBRotationSingleUser/lambda
- Dipendenza: client python InfluxDB 2.0

Amazon Timestream per utenti alternati di InfluxDB

- Nome del modello: SecretsManagerInflux DBRotation MultiUser
- Struttura del **`SecretString`** prevista: the section called "Amazon Timestream per la struttura segreta di InfluxDB".
- Codice sorgente: https://github.com/aws-samples/ aws-secrets-manager-rotation- lambdas/tree/master/SecretsManagerInfluxDBRotationMultiUser/lambda _function.py
- Dipendenza: client python InfluxDB 2.0

Amazon ElastiCache

Per utilizzare questo modello, consulta Rotazione automatica delle password per gli utenti nella Amazon ElastiCache User Guide.

- Nome del modello: SecretsManagerElasticacheUserRotation

- Struttura del **`SecretString`** prevista: the section called "ElastiCache Credenziali Amazon".
- Codice sorgente: https://github.com/aws-samples/ aws-secrets-manager-rotation- lambdas/tree/master/SecretsManagerElasticacheUserRotation/lambda _function.py

Active Directory

Credenziali Active Directory

- Nome del modello: SecretsManagerActiveDirectoryRotationSingleUser
- Struttura del **`SecretString`** prevista: the section called "Credenziali Active Directory".
- Codice sorgente: https://github.com/aws-samples/ aws-secrets-manager-rotation- lambdas/tree/master/SecretsManagerActiveDirectoryRotationSingleUser/lambda _function.py

Tastiera Active Directory

- Nome del modello: SecretsManagerActiveDirectoryAndKeytabRotationSingleUser
- Struttura del **`SecretString`** prevista: the section called "Credenziali Active Directory".
- Codice sorgente: https://github.com/aws-samples/ aws-secrets-manager-rotation- lambdas/tree/master/SecretsManagerActiveDirectoryAndKeytabRotationSingleUser/lambda _function.py
- Dipendenze: msktutil

Other type of secret (Altro tipo di segreto)

Secrets Manager fornisce questo modello come punto di partenza per creare una funzione di rotazione per qualsiasi tipo di segreto.

- Nome del modello: SecretsManagerRotationTemplate
- Codice sorgente: https://github.com/aws-samples/ aws-secrets-manager-rotation- lambdas/tree/master/SecretsManagerRotationTemplate/lambda _function.py

Autorizzazioni del ruolo di esecuzione della funzione di rotazione Lambda per Gestione dei segreti AWS

Infattithe section called "Rotazione tramite funzione Lambda", quando Secrets Manager utilizza una funzione Lambda per ruotare un segreto, Lambda assume un ruolo di esecuzione IAM e fornisce

tali credenziali al codice della funzione Lambda. Per istruzioni su come impostare la rotazione automatica, consulta:

- Rotazione automatica per i segreti del database (console)
- Rotazione automatica per i segreti non relativi al database (console)
- Rotazione automatica (AWS CLI)

Negli esempi seguenti vengono illustrate le policy in linea per i ruoli di esecuzione della funzione di rotazione Lambda. Per creare un ruolo di esecuzione e allegare una policy di autorizzazione, vedere Ruolo di esecuzione di AWS Lambda.

Esempi:

- Policy per un ruolo di esecuzione della funzione di rotazione Lambda
- Istruzione della policy per una chiave gestita dal cliente
- Istruzione della policy per la strategia a utenti alternati

Policy per un ruolo di esecuzione della funzione di rotazione Lambda

La seguente policy di esempio consente alla funzione di rotazione di:

- Esegui le operazioni di Secrets Manager per*SecretARN*.
- Creare una nuova password.
- Impostare la configurazione richiesta se il database o il servizio è in esecuzione in un VPC. Consulta Configurazione di una funzione Lambda per accedere alle risorse in un VPC.

JSON

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

```
                "secretsmanager:UpdateSecretVersionStage"
            ],
            "Resource": "arn:aws:secretsmanager:us-
east-1:123456789012:secret:secretName-AbCdEf"
    },
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetRandomPassword"
            ],
            "Resource": "*"
        },
        {
            "Action": [
                "ec2:CreateNetworkInterface",
                "ec2:DeleteNetworkInterface",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DetachNetworkInterface"
            ],
            "Resource": "*",
            "Effect": "Allow"
        }
    ]
}
```

Istruzione della policy per una chiave gestita dal cliente

Se il segreto è crittografato con una chiave KMS diversa da Chiave gestita da AWS `aws/secretsmanager`, allora è necessario concedere l'autorizzazione per utilizzare la chiave al ruolo di esecuzione Lambda. È possibile utilizzare il contesto di crittografia SecretARN per limitare l'uso della funzione di decrittografia, in modo che il ruolo della funzione di rotazione sia autorizzato ad accedere soltanto per decrittografare il segreto della cui rotazione è responsabile. Nell'esempio seguente viene illustrata un'istruzione da aggiungere alla policy del ruolo di esecuzione per decrittografare il segreto utilizzando la chiave KMS.

```
    {
        "Effect": "Allow",
        "Action": [
            "kms:Decrypt",
            "kms:DescribeKey",
            "kms:GenerateDataKey"
```

```
            ],
            "Resource": "KMSKeyARN",
            "Condition": {
                "StringEquals": {
                    "kms:EncryptionContext:SecretARN": "SecretARN"
                }
            }
        }
```

Per utilizzare la funzione di rotazione per più segreti crittografati con una chiave gestita dal cliente, aggiungi un'istruzione come l'esempio seguente per consentire al ruolo di esecuzione di decrittografare il segreto.

```
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt",
                "kms:DescribeKey",
                "kms:GenerateDataKey"
            ],
            "Resource": "KMSKeyARN",
            "Condition": {
                "StringEquals": {
                    "kms:EncryptionContext:SecretARN": [
                        "arn1",
                        "arn2"
                    ]
                }
            }
        }
```

Istruzione della policy per la strategia a utenti alternati

Per informazioni sulla strategia di rotazione a utenti alternati, consulta la sezione the section called "Strategie di rotazione delle funzioni Lambda".

Per un segreto che contiene le credenziali di Amazon RDS, se utilizzi la strategia degli utenti alternati e il segreto del superutente è gestito da Amazon RDS, devi anche consentire alla funzione di rotazione di effettuare chiamate in sola lettura su APIs Amazon RDS in modo che possa ottenere le informazioni di connessione per il database. Ti consigliamo di allegare la policy AWS gestita Amazon RDSRead OnlyAccess.

La seguente policy di esempio consente alla funzione di:

- Esegui le operazioni di Secrets Manager per*SecretARN*.
- Recupera le credenziali nel segreto del superutente. Gestione dei segreti utilizza le credenziali presenti nel segreto del superutente per aggiornare le credenziali nel segreto che viene ruotato.
- Creare una nuova password.
- Impostare la configurazione richiesta se il database o il servizio è in esecuzione in un VPC. Per ulteriori informazioni, consulta Configurazione di una funzione Lambda per accedere alle risorse in un VPC.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "secretsmanager:UpdateSecretVersionStage"
            ],
            "Resource": "arn:aws:secretsmanager:us-
east-1:123456789012:secret:secretName-AbCdEf"
    },
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue"
            ],
            "Resource": "arn:aws:secretsmanager:us-
east-1:123456789012:secret:secretName-AbCdEf"
    },
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetRandomPassword"
            ],
            "Resource": "*"
```

```
        },
        {
            "Action": [
                "ec2:CreateNetworkInterface",
                "ec2:DeleteNetworkInterface",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DetachNetworkInterface"
            ],
            "Resource": "*",
            "Effect": "Allow"
        }
    ]
}
```

Accesso alla rete per la funzione AWS Lambda di rotazione

Infattithe section called "Rotazione tramite funzione Lambda", quando Secrets Manager utilizza una funzione Lambda per ruotare un segreto, la funzione di rotazione Lambda deve essere in grado di accedere al segreto. Se il segreto contiene credenziali, la funzione Lambda deve anche potere accedere all'origine di tali credenziali, ad esempio un database o un servizio.

Per accedere a un segreto

La funzione di rotazione Lambda deve essere in grado di accedere a un endpoint di Secrets Manager. Se la funzione Lambda può accedere a Internet, è possibile utilizzare un endpoint pubblico. Per trovare un endpoint, consulta the section called "Endpoint di Secrets Manager".

Se la funzione Lambda viene eseguita in un VPC che non dispone di accesso a Internet, si consiglia di configurare gli endpoint privati del servizio Secrets Manager all'interno del VPC. Il tuo VPC può quindi intercettare le richieste indirizzate all'endpoint regionale pubblico e reindirizzarle all'endpoint privato. Per ulteriori informazioni, consulta Endpoint VPC (AWS PrivateLink).

In alternativa, puoi abilitare la funzione Lambda per accedere a un endpoint pubblico di Gestione dei segreti aggiungendo un gateway NAT o un gateway Internet al VPC che consente al traffico dal tuo VPC di raggiungere l'endpoint pubblico Ciò espone il tuo VPC a un livello di rischio perché vi è un indirizzo IP (per il gateway) che può essere soggetto ad attacchi dalla rete Internet pubblica.

(Facoltativo) Per accedere al database o al servizio

Per segreti come le chiavi API, non è necessario aggiornare il database o il servizio di origine insieme al segreto.

Se il tuo database o servizio è in esecuzione su un' EC2 istanza Amazon in un VPC, ti consigliamo di configurare la funzione Lambda per l'esecuzione nello stesso VPC. Quindi la funzione di rotazione può comunicare direttamente con il servizio. Per ulteriori informazioni, consulta Configurazione dell'accesso VPC.

Per consentire alla funzione Lambda di accedere al database o al servizio, è necessario assicurarsi che i gruppi di sicurezza collegati alla funzione di rotazione Lambda consentano connessioni in uscita al database o al servizio. Inoltre, è necessario accertarsi che i gruppi di sicurezza collegati al database o al servizio consentano le connessioni in entrata dalla funzione di rotazione Lambda.

Risolvi i problemi Gestione dei segreti AWS di rotazione

Per molti servizi, Secrets Manager utilizza una funzione Lambda per ruotare i segreti. Per ulteriori informazioni, consulta the section called "Rotazione tramite funzione Lambda". La funzione di rotazione Lambda interagisce con il database o il servizio a cui appartiene il segreto e con Gestione dei segreti. Quando la rotazione non funziona come previsto, è necessario innanzitutto controllare i CloudWatch registri.

> **Note**
>
> Alcuni servizi possono gestire i segreti per te, tra cui la gestione della rotazione automatica. Per ulteriori informazioni, consulta the section called "Rotazione gestita".

Argomenti

- Come risolvere gli errori di rotazione segreti nelle funzioni AWS Lambda
- Nessuna attività dopo "Found credentials in environment variables" (Trovate credenziali nelle variabili di ambiente)
- Nessuna attività dopo "createSecret"
- Errore: "Access to KMS is not allowed"
- Errore: "Key is missing from secret JSON" (Chiave non presente nella struttura JSON del segreto)

- Errore: "setSecret: Unable to log into database" (setSecret: impossibile accedere al database)
- Errore: "Unable to import module 'lambda_function'"
- Aggiornare una funzione di rotazione esistente da Python 3.7 a 3.9
- Aggiorna una funzione di rotazione esistente da Python 3.9 a 3.10
- AWS Lambda rotazione segreta con errore PutSecretValue
- Errore: «Errore durante l'esecuzione di lambda durante il passaggio<arn>» <a rotation>

Come risolvere gli errori di rotazione segreti nelle funzioni AWS Lambda

Se riscontri errori di rotazione segreti con le funzioni Lambda, utilizza i seguenti passaggi per risolvere il problema.

Possibili cause

- Esecuzioni simultanee insufficienti per la funzione Lambda
- Condizioni di gara dovute a più chiamate API durante la rotazione
- Logica della funzione Lambda errata
- Problemi di rete tra la funzione Lambda e il database

Procedure generali per la risoluzione

1. Analizza CloudWatch i log:
   - Cerca messaggi di errore specifici o comportamenti imprevisti nei log delle funzioni Lambda
   - Verifica che tutti i passaggi di rotazione (CreateSecret,, SetSecretTestSecret,FinishSecret) siano stati tentati
2. Controlla le chiamate API durante la rotazione:
   - Evita di effettuare chiamate API mutanti sul segreto durante la rotazione Lambda
   - Assicurati che non vi siano condizioni di gara tra le chiamate e RotateSecret PutSecretValue
3. Verifica la logica della funzione Lambda:
   - Conferma di utilizzare il codice di AWS esempio più recente per la rotazione segreta
   - Se utilizzi codice personalizzato, controllalo per una corretta gestione di tutte le fasi di rotazione

4. Controlla la configurazione di rete:
   - Verifica che le regole del gruppo di sicurezza consentano alla funzione Lambda di accedere al database
   - Garantire l'accesso corretto agli endpoint VPC o agli endpoint pubblici per Secrets Manager
5. Prova le versioni segrete:
   - Verifica che la AWSCURRENT versione del segreto consenta l'accesso al database
   - Controlla se AWSPENDING le AWSPREVIOUS nostre versioni sono valide
6. Cancella le rotazioni in sospeso:
   - Se la rotazione fallisce costantemente, cancella l'etichetta della AWSPENDING fase temporanea e riprova a ruotare
7. Controlla le impostazioni di concorrenza Lambda:
   - Verifica che le impostazioni di concorrenza siano appropriate per il tuo carico di lavoro
   - Se sospetti problemi di concorrenza, consulta la sezione «Risoluzione dei problemi di rotazione legati alla concorrenza»

Nessuna attività dopo "Found credentials in environment variables" (Trovate credenziali nelle variabili di ambiente)

Se non si verifica alcuna attività dopo la visualizzazione del messaggio "Found credentials in environment variables" (Trovate credenziali nelle variabili di ambiente) e la durata del processo è lunga (ad esempio, il timeout predefinito di Lambda è 30.000 ms), è possibile che il timeout della funzione Lambda si verifichi durante il tentativo di raggiungere l'endpoint di Gestione dei segreti.

La funzione di rotazione Lambda deve essere in grado di accedere a un endpoint di Secrets Manager. Se la funzione Lambda può accedere a Internet, è possibile utilizzare un endpoint pubblico. Per trovare un endpoint, consulta the section called “Endpoint di Secrets Manager”.

Se la funzione Lambda viene eseguita in un VPC che non dispone di accesso a Internet, si consiglia di configurare gli endpoint privati del servizio Secrets Manager all'interno del VPC. Il tuo VPC può quindi intercettare le richieste indirizzate all'endpoint regionale pubblico e reindirizzarle all'endpoint privato. Per ulteriori informazioni, consulta Endpoint VPC (AWS PrivateLink).

In alternativa, puoi abilitare la funzione Lambda per accedere a un endpoint pubblico di Gestione dei segreti aggiungendo un gateway NAT o un gateway Internet al VPC che consente al traffico dal tuo

VPC di raggiungere l'endpoint pubblico Ciò espone il tuo VPC a un livello di rischio perché vi è un indirizzo IP (per il gateway) che può essere soggetto ad attacchi dalla rete Internet pubblica.

Nessuna attività dopo "createSecret"

Di seguito sono riportati i problemi che possono causare l'interruzione della rotazione dopo createSecret:

La rete VPC ACLs non consente l'ingresso e l'uscita del traffico HTTPS.

> Per ulteriori informazioni, consulta Controllare il traffico verso le sottoreti utilizzando la rete ACLs nella Amazon VPC User Guide.

La configurazione del timeout della funzione Lambda è troppo breve per eseguire il processo.

> Per ulteriori informazioni, consulta Configurazione delle opzioni della funzione Lambda nella Guida per gli sviluppatori di AWS Lambda .

L'endpoint VPC di Secrets Manager non consente al VPC di entrare CIDRs nei gruppi di sicurezza assegnati.

> Per ulteriori informazioni, consulta Controllo del traffico verso le risorse utilizzando gruppi di sicurezza nella Guida per l'utente di Amazon VPC.

La policy degli endpoint VPC di Gestione dei segreti non consente a Lambda di utilizzare l'endpoint VPC.

> Per ulteriori informazioni, consulta the section called “Endpoint VPC (AWS PrivateLink)”.

Il segreto utilizza la rotazione alternata degli utenti, il segreto superutente è gestito da Amazon RDS e la funzione Lambda non può accedere all'API RDS.

> Per la rotazione alternata degli utenti in cui il segreto del superutente è gestito da un altro servizio AWS, la funzione di rotazione Lambda deve essere in grado di chiamare l'endpoint per ottenere le informazioni sulla connessione al database. Si consiglia di configurare un endpoint VPC per il servizio del database. Per ulteriori informazioni, consulta:
>
> - Endpoint VPC dell'interfaccia e API Amazon RDS nella Guida per l'utente di Amazon RDS.
> - Utilizzo degli endpoint VPC nella Guida di gestione di Amazon Redshift.

Errore: "Access to KMS is not allowed"

Se ricevi l'errore `ClientError: An error occurred (AccessDeniedException) when calling the GetSecretValue operation: Access to KMS is not allowed`, la funzione

di rotazione non è autorizzata a decrittografare il segreto utilizzando la chiave KMS utilizzata per crittografare il segreto. La policy delle autorizzazioni potrebbe contenere una condizione che limita il contesto di crittografia a un segreto specifico. Per informazioni sulle autorizzazioni richieste, consulta la sezione the section called "Istruzione della policy per una chiave gestita dal cliente".

Errore: "Key is missing from secret JSON" (Chiave non presente nella struttura JSON del segreto)

Una funzione di rotazione Lambda richiede che il valore segreto si trovi in una struttura JSON specifica. Se viene visualizzato questo errore, è possibile che la chiave a cui la funzione di rotazione ha cercato di accedere non sia presente nella struttura JSON. Per informazioni sulla struttura JSON per ogni tipo di segreto, consulta the section called "Struttura JSON di un segreto ".

Errore: "setSecret: Unable to log into database" (setSecret: impossibile accedere al database)

Di seguito sono riportati i problemi che possono causare questo errore:

La funzione di rotazione non può accedere al database.

- Se la durata del processo è lunga, ad esempio oltre 5.000 ms, la funzione di rotazione Lambda potrebbe non essere in grado di accedere al database tramite la rete.

  Se il database o il servizio è in esecuzione su un'istanza Amazon EC2 in un VPC, è consigliabile configurare la funzione Lambda in modo che sia eseguita nello stesso VPC. Quindi la funzione di rotazione può comunicare direttamente con il servizio. Per ulteriori informazioni, consulta Configurazione dell'accesso VPC.

  Per consentire alla funzione Lambda di accedere al database o al servizio, è necessario assicurarsi che i gruppi di sicurezza collegati alla funzione di rotazione Lambda consentano connessioni in uscita al database o al servizio. Inoltre, è necessario accertarsi che i gruppi di sicurezza collegati al database o al servizio consentano le connessioni in entrata dalla funzione di rotazione Lambda.

Le credenziali nel segreto non sono corrette.

- Se la durata del processo è breve, la funzione di rotazione Lambda potrebbe non essere in grado di autenticarsi con le credenziali segrete. Verifica le credenziali accedendo manualmente con le informazioni contenute nelle `AWSPREVIOUS` versioni `AWSCURRENT` e nelle versioni del segreto utilizzando il comando. AWS CLI `get-secret-value`

Il database utilizza **`scram-sha-256`** per crittografare le password.

Se il database è Aurora PostgreSQL versione 13 o successiva e questo utilizza `scram-sha-256` per crittografare le password, ma la funzione di rotazione utilizza `libpq` versione 9 o precedente che non supporta `scram-sha-256`, in questo caso la funzione di rotazione non può connettersi al database.

Per determinare quali utenti del database utilizzano la crittografia **`scram-sha-256`**

- Consulta Checking for users with non-SCRAM passwords (Verifica della presenza di utenti con password non SCRAM) nel blog Autenticazione SCRAM in RDS per PostgreSQL 13.

Per determinare quale versione di **`libpq`** viene utilizzata dalla funzione di rotazione

1. In un computer basato su Linux, sulla console Lambda, accedi alla funzione di rotazione e scarica il pacchetto di implementazione. Decomprimi il file zip in una directory di lavoro.
2. Nella riga di comando, nella directory di lavoro, esegui:

   ```
   readelf -a libpq.so.5 | grep RUNPATH
   ```
3. Se vedi la stringa *`PostgreSQL-9.4.x`*, o qualsiasi versione principale inferiore a 10, la funzione di rotazione non supporta `scram-sha-256`.
   - Output per una funzione di rotazione che non supporta `scram-sha-256`:

     ```
     0x000000000000001d (RUNPATH) Library runpath: [/
     local/p4clients/pkgbuild-a1b2c/workspace/build/
     PostgreSQL/PostgreSQL-9.4.x_client_only.123456.0/AL2_x86_64/
     DEV.STD.PTHREAD/build/private/tmp/brazil-path/build.libfarm/lib:/
     local/p4clients/pkgbuild-a1b2c/workspace/src/PostgreSQL/build/
     private/install/lib]
     ```
   - Output per una funzione di rotazione che supporta `scram-sha-256`:

     ```
     0x000000000000001d (RUNPATH) Library runpath: [/
     local/p4clients/pkgbuild-a1b2c/workspace/build/
     PostgreSQL/PostgreSQL-10.x_client_only.123456.0/AL2_x86_64/
     DEV.STD.PTHREAD/build/private/tmp/brazil-path/build.libfarm/lib:/
     local/p4clients/pkgbuild-a1b2c/workspace/src/PostgreSQL/build/
     private/install/lib]
     ```

- Output per una funzione di rotazione che supporta `scram-sha-256`:

  ```
  0x000000000000001d (RUNPATH) Library runpath: [/local/
  p4clients/pkgbuild- a1b2c /workspace/build/PostgreSQL/
  PostgreSQL-14.x_client_only. 123456 .0/AL2_x86_64/
  DEV.STD.PTHREAD/build/private/tmp/brazil-path/build.libfarm/lib:/
  local/p4clients/pkgbuild- a1b2c /workspace/src/PostgreSQL/build/
  private/install/lib]
  ```

- Output per una funzione di rotazione che supporta `scram-sha-256`:

  ```
  0x000000000000001d (RUNPATH) Library runpath: [/local/p4clients/
  pkgbuild- a1b2c/workspace/build/PostgreSQL/PostgreSQL-
  14.x_client_only.123456.0/AL2_x86_64/DEV.STD.PTHREAD/build/
  private/tmp/brazil- path/build.libfarm/lib:/local/p4clients/
  pkgbuild- a1b2c/workspace/src/PostgreSQL/build/private/install/
  lib]
  ```

> **Note**
>
> Se imposti la rotazione segreta automatica prima del 30 dicembre 2021, la funzione di rotazione inclusa in una versione precedente non la `libpq` supportava. `scram-sha-256` Per supportare `scram-sha-256`, è necessario ricreare la funzione di rotazione.

Il database richiede l' SSL/TLS accesso.

Se il database richiede una SSL/TLS connessione, ma la funzione di rotazione utilizza una connessione non crittografata, la funzione di rotazione non può connettersi al database. Le funzioni di rotazione per Amazon RDS (tranne Oracle e Db2) e Amazon DocumentDB usano automaticamente Secure Socket Layer (SSL) o Transport Layer Security (TLS) per connettersi al database, se disponibile. Altrimenti usano una connessione non crittografata.

> **Note**
>
> Se hai impostato la rotazione segreta automatica prima del 20 dicembre 2021, la tua funzione di rotazione potrebbe essere basata su un modello precedente che non

supportavaSSL/TLS. To support connections that use SSL/TLS, devi ricreare la funzione di rotazione.

Per determinare quando è stata creata la funzione di rotazione

1. Nella console Secrets Manager https://console.aws.amazon.com/secretsmanager/, apri il tuo segreto. Nella sezione Rotation configuration (Configurazione rotazione), sotto Lambda rotation function (Funzione di rotazione Lambda), viene visualizzato l'ARN della funzione Lambda, ad esempio, `arn:aws:lambda:aws-region:123456789012:function:SecretsManagerMyRotationFunction`. Copia il nome della funzione dalla fine dell'ARN, in questo esempio `SecretsManagerMyRotationFunction` .
2. Nella AWS Lambda console https://console.aws.amazon.com/lambda/, in Funzioni, incolla il nome della funzione Lambda nella casella di ricerca, scegli Invio, quindi scegli la funzione Lambda.
3. Nella pagina dei dettagli della funzione, nella scheda Configuration (Configurazione), in Tag, copia il valore accanto alla chiave aws:cloudformation:stack-name.
4. Nella AWS CloudFormation console https://console.aws.amazon.com/cloudformation, in Stacks, incolla il valore della chiave nella casella di ricerca, quindi scegli Invio.
5. L'elenco degli stack filtra in modo che venga visualizzato solo lo stack che ha creato la funzione di rotazione Lambda. Nella colonna Created date (Data di creazione), visualizza la data di creazione dello stack. Questa è la data di creazione della funzione di rotazione Lambda.

Errore: "Unable to import module 'lambda_function'"

Potresti ricevere questo errore se stai eseguendo una funzione Lambda precedente che è stata aggiornata automaticamente da Python 3.7 a una versione più recente di Python. Per risolvere l'errore, puoi modificare la versione della funzione Lambda in Python 3.7 e quindi the section called "Aggiornare una funzione di rotazione esistente da Python 3.7 a 3.9". Per ulteriori informazioni, consulta la sezione Perché la rotazione della mia funzione Lambda di Secrets Manager non è riuscita con un errore "pg module not found"? in AWS re:Post.

Aggiornare una funzione di rotazione esistente da Python 3.7 a 3.9

Alcune funzioni di rotazione create prima di novembre 2022 utilizzavano Python 3.7. L' AWS SDK per Python ha smesso di supportare Python 3.7 a dicembre 2023. Per ulteriori informazioni, consulta Aggiornamenti della politica di supporto di Python per AWS SDKs e strumenti. Per passare a una nuova funzione di rotazione che utilizza Python 3.9, puoi aggiungere una proprietà runtime a una funzione di rotazione esistente o ricreare la funzione di rotazione.

Per scoprire quali funzioni di rotazione Lambda usano Python 3.7

1. Accedi a Console di gestione AWS e apri la AWS Lambda console all'indirizzo https://console.aws.amazon.com/lambda/.
2. Nell'elenco delle funzioni, filtra per **`SecretsManager`**.
3. Nell'elenco filtrato delle funzioni, in Runtime, cerca Python 3.7.

Per eseguire l'aggiornamento a Python 3.9:

- Opzione 1: ricrea la funzione di rotazione usando CloudFormation
- Opzione 2: aggiorna il runtime per la funzione di rotazione esistente utilizzando CloudFormation
- Opzione 3: per AWS CDK gli utenti, aggiornate la libreria CDK

Opzione 1: ricrea la funzione di rotazione usando CloudFormation

Quando si utilizza la console Secrets Manager per attivare la rotazione, Secrets Manager crea CloudFormation le risorse necessarie, inclusa la funzione di rotazione Lambda. Se hai utilizzato la console per attivare la rotazione o hai creato la funzione di rotazione utilizzando una CloudFormation pila, puoi utilizzare lo stesso CloudFormation stack per ricreare la funzione di rotazione con un nuovo nome. La nuova funzione utilizza la versione più recente di Python.

Per trovare lo CloudFormation stack che ha creato la funzione di rotazione

- Nella pagina dei dettagli della funzione Lambda, nella sezione Configurazione scegli Tag. Visualizza l'ARN accanto a aws:cloudformation:stack-id.

  Il nome dello stack è incorporato nell'ARN, come illustrato nell'esempio seguente.

  - ARN: `arn:aws:cloudformation:us-west-2:408736277230:stack/`**`SecretsManagerRDSMySQLRotationSingleUser5c2-SecretRotationScheduleHostedRotationLambda`**`-3CUDHZMDMBO8/79fc9050-2eef-11ed-`

- Nome stack: **SecretsManagerRDSMySQLRotationSingleUser5c2-SecretRotationScheduleHostedRotationLambda**

Per ricreare una funzione di rotazione (CloudFormation)

1. In CloudFormation, cerca lo stack per nome, quindi scegli Aggiorna.

   Se viene visualizzata una finestra di dialogo che consiglia di aggiornare lo stack principale, scegli Vai allo stack principale, quindi scegli Aggiorna.
2. Nella pagina Update stack, in Prepara modello, scegli Modifica in Application Composer, quindi in Modifica modello in Application Composer, scegli il pulsante Modifica in Application Composer.
3. In Application Composer, effettuate le seguenti operazioni:

   a. Nel codice del modello, in`SecretRotationScheduleHostedRotationLambda`, sostituisci il valore di `"functionName": "SecretsManagerTestRotationRDS"` con un nuovo nome di funzione, ad esempio in JSON, **"functionName": "SecretsManagerTestRotationRDSupdated"**

   b. Scegli Aggiorna modello.

   c. Nella CloudFormation finestra di dialogo Continua a, scegli Conferma e continua con CloudFormation.
4. Continua con il flusso di lavoro CloudFormation dello stack, quindi scegli Invia.

Opzione 2: aggiorna il runtime per la funzione di rotazione esistente utilizzando CloudFormation

Quando si utilizza la console Secrets Manager per attivare la rotazione, Secrets Manager crea CloudFormation le risorse necessarie, inclusa la funzione di rotazione Lambda. Se hai utilizzato la console per attivare la rotazione o hai creato la funzione di rotazione utilizzando uno CloudFormation stack, puoi utilizzare lo stesso CloudFormation stack per aggiornare il runtime della funzione di rotazione.

Per trovare lo CloudFormation stack che ha creato la funzione di rotazione

- Nella pagina dei dettagli della funzione Lambda, nella sezione Configurazione scegli Tag. Visualizza l'ARN accanto a aws:cloudformation:stack-id.

  Il nome dello stack è incorporato nell'ARN, come illustrato nell'esempio seguente.

- ARN: `arn:aws:cloudformation:us-west-2:408736277230:stack/`**`SecretsManagerRDSMySQLRotationSingleUser5c2-SecretRotationScheduleHostedRotationLambda`**`-3CUDHZMDMBO8/79fc9050-2eef-11ed-`
- Nome stack: **`SecretsManagerRDSMySQLRotationSingleUser5c2-SecretRotationScheduleHostedRotationLambda`**

Per aggiornare il runtime per una funzione di rotazione (CloudFormation)

1. In CloudFormation, cerca lo stack per nome, quindi scegli Aggiorna.

   Se viene visualizzata una finestra di dialogo che consiglia di aggiornare lo stack principale, scegli Vai allo stack principale, quindi scegli Aggiorna.
2. Nella pagina Update stack, in Prepara modello, scegli Modifica in Application Composer, quindi in Modifica modello in Application Composer, scegli il pulsante Modifica in Application Composer.
3. In Application Composer, effettuate le seguenti operazioni:

   a. Nel modello JSON, per aggiungere`SecretRotationScheduleHostedRotationLambda`, sotto `PropertiesParameters`, sotto. **`"runtime": "python3.9"`**

   b. Scegli Aggiorna modello.

   c. Nella CloudFormation finestra di dialogo Continua a, scegli Conferma e continua con CloudFormation.
4. Continua con il flusso di lavoro CloudFormation dello stack, quindi scegli Invia.

Opzione 3: per AWS CDK gli utenti, aggiornate la libreria CDK

Se hai utilizzato la versione AWS CDK precedente alla v2.94.0 per impostare la rotazione per il tuo segreto, puoi aggiornare la funzione Lambda eseguendo l'aggiornamento alla versione 2.94.0 o successiva. Per ulteriori informazioni, consulta la Guida per gli sviluppatori v2 di AWS Cloud Development Kit (AWS CDK).

Aggiorna una funzione di rotazione esistente da Python 3.9 a 3.10

Secrets Manager sta passando da Python 3.9 a 3.10 per le funzioni di rotazione Lambda. Per passare a una nuova funzione di rotazione che utilizza Python 3.10, dovrai seguire il percorso di aggiornamento in base al tuo metodo di distribuzione. Usa le seguenti procedure per aggiornare sia la versione di Python che le dipendenze sottostanti.

Per scoprire quali funzioni di rotazione Lambda usano Python 3.9

1. Accedi a Console di gestione AWS e apri la console all' AWS Lambda indirizzo. https://console.aws.amazon.com/lambda/
2. Nell'elenco delle funzioni, filtra per **`SecretsManager`**.
3. Nell'elenco filtrato delle funzioni, in Runtime, cerca**`Python 3.9`**.

Aggiorna i percorsi in base al metodo di distribuzione

Le funzioni di rotazione Lambda identificate in questo elenco possono essere distribuite tramite la console Secrets Manager, AWS Serverless Application Repository le app o le trasformazioni. CloudFormation Ciascuna di queste strategie di distribuzione ha un percorso di aggiornamento distinto.

Utilizza una delle seguenti procedure per aggiornare le funzioni di rotazione Lambda, a seconda di come è stata distribuita la funzione.

Gestione dei segreti AWS console-deployed functions

Una nuova funzione Lambda deve essere distribuita tramite Gestione dei segreti AWS console poiché non è possibile aggiornare manualmente le dipendenze per le funzioni Lambda esistenti.

Usa la seguente procedura per aggiornare le funzioni distribuite dalla console. Gestione dei segreti AWS

1. Apri la console Secrets Manager all'indirizzo https://console.aws.amazon.com/secretsmanager/.
2. In Gestione dei segreti AWS, seleziona Segreti. Seleziona il segreto che utilizza la funzione Lambda che desideri aggiornare.
3. Vai alla scheda Rotazioni e seleziona l'opzione Aggiorna le configurazioni di rotazione.
4. In Funzioni di rotazione, scegli Crea una nuova funzione e inserisci un nuovo nome per la funzione di rotazione Lambda.

    a. (Facoltativo) Una volta completato l'aggiornamento, puoi testare la funzione Lambda aggiornata per confermare che funzioni come previsto. Nella scheda Rotazione, selezionate Rotate Secret Immediatamente per avviare una rotazione immediata.

    b. (Facoltativo) Puoi visualizzare i log delle funzioni e la versione di Python utilizzata in fase di esecuzione in Amazon. CloudWatch Per ulteriori informazioni, consulta

Visualizzazione dei CloudWatch registri per le funzioni Lambda nella Guida per AWS Lambda gli sviluppatori.

5. Una volta impostata la nuova funzione di rotazione, è possibile eliminare la vecchia funzione di rotazione.

AWS Serverless Application Repository deployments

La procedura seguente mostra come aggiornare le AWS Serverless Application Repository distribuzioni. Le funzioni Lambda distribuite tramite AWS Serverless Application Repository hanno un banner `This function belongs to an application. Click here to manage it.` che indica che include un collegamento all'applicazione Lambda a cui appartiene la funzione.

> **Important**
>
> AWS Serverless Application Repository la disponibilità dipende. Regione AWS

Utilizzare la procedura seguente per aggiornare le funzioni AWS Serverless Application Repository distribuite.

1. Aprire la AWS Lambda console all'indirizzo https://console.aws.amazon.com/lambda/.
2. Vai alla scheda Configurazioni della funzione Lambda che deve essere aggiornata.
   - Avrai bisogno delle seguenti informazioni sulla tua funzione per aggiornare l'applicazione AWS Serverless Application Repository distribuita. Puoi trovare queste informazioni nella console Lambda.
     - Nome dell'applicazione Lambda
       - Il nome dell'applicazione Lambda può essere trovato utilizzando il link nel banner. Ad esempio, il banner riporta quanto segue`serverlessrepo-`*`SecretsManagerRedshiftRotationSingleUser`*. Il nome in questo esempio è`SecretsManagerRedshiftRotationSingleUser`.
     - Nome della funzione di rotazione Lambda
     - Endpoint Secrets Manager
       - L'endpoint è disponibile nelle schede Configurazioni e Variabili di ambiente assegnate alla variabile SECRETS_MANAGER_ENDPOINT.

3. Per aggiornare Python, è necessario aggiornare la versione semantica dell'applicazione serverless. Vedi Aggiornamento delle applicazioni nella Guida per gli sviluppatori.AWS Serverless Application Repository

Custom Lambda rotation functions

Se hai creato funzioni di rotazione Lambda personalizzate, dovrai aggiornare le dipendenze e i runtime di ogni pacchetto per queste funzioni. Per ulteriori informazioni, consulta Aggiornamento del runtime della funzione Lambda alla versione più recente.

AWS::SecretsManager-2024-09-16 transform macro

Se la funzione Lambda viene distribuita tramite questa trasformazione, l'aggiornamento degli stack utilizzando il modello esistente consentirà di utilizzare il runtime Lambda aggiornato.

Utilizza la seguente procedura per aggiornare lo CloudFormation stack utilizzando il modello esistente.

1. Apri la CloudFormation console in https://console.aws.amazon.com/cloudformation.
2. Nella pagina Stacks, seleziona lo stack che desideri aggiornare.
3. Scegli Aggiorna nel riquadro dei dettagli dello stack.
4. Per Scegli un metodo di aggiornamento del modello, seleziona Aggiornamento diretto.
5. Nella pagina Specificare il modello, seleziona Usa modello esistente.
6. Mantieni tutte le altre opzioni ai valori predefiniti, quindi scegli Aggiorna stack.

Se riscontri problemi durante l'aggiornamento dello stack, consulta Determinare la causa di un errore dello stack nella Guida per l'CloudFormation utente.

AWS::SecretsManager-2020-07-23 transform macro

Ti consigliamo di migrare alla versione di trasformazione più recente, se stai utilizzando. `AWS::SecretsManager-2020-07-23` Per ulteriori informazioni, consulta Introduzione a una versione migliorata della Gestione dei segreti AWS trasformazione: AWS:: SecretsManager -2024-09-16 nel Security Blog.AWS Se continui a utilizzare`AWS::SecretsManager-2020-07-23`, potresti riscontrare un errore di mancata corrispondenza tra la tua versione di runtime e gli artefatti del codice della funzione Lambda. Per ulteriori informazioni, vedi AWS::SecretsManager: RotationSchedule HostedRotationLambda nel modello di riferimento.CloudFormation

Se riscontri problemi durante l'aggiornamento dello stack, determina la causa di un errore dello stack nella Guida per l'CloudFormation utente.

Verifica l'aggiornamento di Python

Per verificare l'aggiornamento di Python, apri la console Lambda (https://console.aws.amazon.com/lambda/) e accedi alla pagina Function. Seleziona la funzione che hai aggiornato. Nella sezione Codice sorgente, esamina i file inclusi nella directory e assicurati che la versione del file.so di Python sia valida. `3.10`

AWS Lambda rotazione segreta con errore **`PutSecretValue`**

Se utilizzi un ruolo presunto o una rotazione tra account con Secrets Manager e trovi un RotationFailed evento nel CloudTrail messaggio: Pending secret version *`VERSION_ID`* for Secret *`SECRET_ARN`* was not created by Lambda *`LAMBDA_ARN`*. Rimuovi l'etichetta `AWSPENDING` staging e riavvia la rotazione, quindi devi aggiornare la funzione Lambda per utilizzare il parametro. `RotationToken`

Aggiorna la funzione di rotazione Lambda per includere **`RotationToken`**

1. Scarica il codice della funzione Lambda
   - Apri la console Lambda
   - Nel pannello di navigazione, scegli Funzioni
   - Seleziona la tua funzione di rotazione segreta Lambda per il nome della funzione
   - Per Download, scegli una delle seguenti opzioni: codice funzione .zip, AWS SAM file, Entrambi
   - Scegli OK per salvare la funzione sul tuo computer locale.
2. Modifica `Lambda_handler`

   Includi il parametro rotation_token nel passaggio create_secret per la rotazione tra account:

```
def lambda_handler(event, context):
    """Secrets Manager Rotation Template

    This is a template for creating an AWS Secrets Manager rotation lambda

    Args:
```

```
        event (dict): Lambda dictionary of event parameters. These keys must
include the following:
            - SecretId: The secret ARN or identifier
            - ClientRequestToken: The ClientRequestToken of the secret version
            - Step: The rotation step (one of createSecret, setSecret, testSecret,
or finishSecret)
            - RotationToken: the rotation token to put as parameter for
PutSecretValue call

        context (LambdaContext): The Lambda runtime information

    Raises:
        ResourceNotFoundException: If the secret with the specified arn and stage
does not exist

        ValueError: If the secret is not properly configured for rotation

        KeyError: If the event parameters do not contain the expected keys

    """
    arn = event['SecretId']
    token = event['ClientRequestToken']
    step = event['Step']
    # Add the rotation token
    rotation_token = event['RotationToken']

    # Setup the client
    service_client = boto3.client('secretsmanager',
endpoint_url=os.environ['SECRETS_MANAGER_ENDPOINT'])

    # Make sure the version is staged correctly
    metadata = service_client.describe_secret(SecretId=arn)
    if not metadata['RotationEnabled']:
        logger.error("Secret %s is not enabled for rotation" % arn)
        raise ValueError("Secret %s is not enabled for rotation" % arn)
    versions = metadata['VersionIdsToStages']
    if token not in versions:
        logger.error("Secret version %s has no stage for rotation of secret %s." %
(token, arn))
        raise ValueError("Secret version %s has no stage for rotation of secret
%s." % (token, arn))
    if "AWSCURRENT" in versions[token]:
        logger.info("Secret version %s already set as AWSCURRENT for secret %s." %
(token, arn))
```

```
        return
    elif "AWSPENDING" not in versions[token]:
        logger.error("Secret version %s not set as AWSPENDING for rotation of
 secret %s." % (token, arn))
        raise ValueError("Secret version %s not set as AWSPENDING for rotation of
 secret %s." % (token, arn))
    # Use rotation_token
    if step == "createSecret":
        create_secret(service_client, arn, token, rotation_token)

    elif step == "setSecret":
        set_secret(service_client, arn, token)

    elif step == "testSecret":
        test_secret(service_client, arn, token)

    elif step == "finishSecret":
        finish_secret(service_client, arn, token)

    else:
        raise ValueError("Invalid step parameter")
```

3. Modifica codice `create_secret`

   Modifica la `create_secret` funzione per accettare e utilizzare il `rotation_token` parametro:

```
# Add rotation_token to the function
def create_secret(service_client, arn, token, rotation_token):
"""Create the secret

This method first checks for the existence of a secret for the passed in token. If
 one does not exist, it will generate a
new secret and put it with the passed in token.

Args:
service_client (client): The secrets manager service client

arn (string): The secret ARN or other identifier

token (string): The ClientRequestToken associated with the secret version
```

```
rotation_token (string): the rotation token to put as parameter for PutSecretValue
 call

Raises:
ResourceNotFoundException: If the secret with the specified arn and stage does not
 exist

"""
# Make sure the current secret exists
service_client.get_secret_value(SecretId=arn, VersionStage="AWSCURRENT")

# Now try to get the secret version, if that fails, put a new secret
try:
service_client.get_secret_value(SecretId=arn, VersionId=token,
 VersionStage="AWSPENDING")
logger.info("createSecret: Successfully retrieved secret for %s." % arn)
except service_client.exceptions.ResourceNotFoundException:
# Get exclude characters from environment variable
exclude_characters = os.environ['EXCLUDE_CHARACTERS'] if 'EXCLUDE_CHARACTERS' in
 os.environ else '/@"\'\\'
# Generate a random password
passwd = service_client.get_random_password(ExcludeCharacters=exclude_characters)

# Put the secret, using rotation_token
service_client.put_secret_value(SecretId=arn, ClientRequestToken=token,
 SecretString=passwd['RandomPassword'], VersionStages=['AWSPENDING'],
 RotationToken=rotation_token)
logger.info("createSecret: Successfully put secret for ARN %s and version %s." %
 (arn, token))
```

4. Carica il codice della funzione Lambda aggiornato

   Dopo aver aggiornato il codice della funzione Lambda, caricalo per ruotare il tuo segreto.

Errore: «Errore durante l'esecuzione di lambda durante il passaggio*<arn>*» *<a rotation>*

Se si verificano errori intermittenti di rotazione segreta e la funzione Lambda rimane bloccata in un ciclo di set, ad esempio tra CreateSecret eSetSecret, il problema potrebbe essere correlato alle impostazioni di concorrenza.

Passaggi per la risoluzione dei problemi di concorrenza

> **Warning**
>
> L'impostazione del parametro di concorrenza assegnato su un valore inferiore a 10 può causare una limitazione a causa di thread di esecuzione insufficienti per la funzione Lambda. Per ulteriori informazioni, consulta Understanding Reserved Concurrency e Provisioned Concurrency nella Developer Guide. AWS Lambda AWS Lambda

1. Controlla e modifica le impostazioni di concorrenza Lambda:
   - Verifica che non `reserved_concurrent_executions` sia impostato su un valore troppo basso (ad esempio, 1)
   - Se utilizzate la concorrenza riservata, impostatela su almeno 10
   - Prendi in considerazione l'utilizzo di una concorrenza senza riserve per una maggiore flessibilità
2. Per la concorrenza preimpostata:
   - Non impostare esplicitamente il parametro di concorrenza fornito (ad esempio, in Terraform).
   - Se devi impostarlo, usa un valore di almeno 10.
   - Esegui un test accurato per assicurarti che il valore scelto sia adatto al tuo caso d'uso.
3. Monitora e regola la concorrenza:
   - Calcola la concorrenza utilizzando questa formula: Concorrenza = (media delle richieste al secondo) * (durata media della richiesta in secondi). Per ulteriori informazioni, vedere Stima della concorrenza riservata.
   - Osserva e registra i valori durante le rotazioni per determinare le impostazioni di concorrenza appropriate.
   - Fate attenzione quando impostate valori di concorrenza bassi. Possono causare un throttling se non ci sono abbastanza thread di esecuzione disponibili.

Per ulteriori informazioni sulla configurazione della concorrenza Lambda, consulta Configuring reserved concurrency e Configuring provisioned concurrency nella Developer Guide. AWS Lambda

Pianificazioni di rotazione

Secrets Manager ruota il tuo segreto in base a una pianificazione durante una finestra di rotazione da te impostata. Per impostare la pianificazione e la finestra, utilizzate un'espressione cron () o rate () insieme alla durata della finestra. Secrets Manager ruota il tuo segreto in qualsiasi momento durante la finestra di rotazione. Potete ruotare un segreto ogni quattro ore all'interno di una finestra di rotazione di appena un'ora.

Per attivare la rotazione, consulta:

- the section called “Rotazione gestita”
- the section called “Rotazione automatica per i segreti del database (console)”
- the section called “Rotazione automatica per i segreti non relativi al database (console)”

I programmi di rotazione di Secrets Manager utilizzano il fuso orario UTC.

Finestre di rotazione

Una finestra di rotazione di Secrets Manager è simile a una finestra di manutenzione. Puoi impostare la finestra di rotazione quando vuoi che il tuo segreto ruoti e Secrets Manager ruota il tuo segreto in qualche momento durante la finestra di rotazione.

Le finestre di rotazione di Secrets Manager iniziano sempre all'ora. Per un programma di rotazione che utilizza un'`rate( )`espressione in giorni, la finestra di rotazione inizia a mezzanotte. È possibile impostare l'ora di inizio della finestra di rotazione utilizzando un'`cron( )`espressione. Per alcuni esempi, consulta the section called “Espressioni Cron”.

Per impostazione predefinita, la finestra di rotazione si chiude dopo un'ora per un programma di rotazione in ore e alla fine della giornata per un programma di rotazione in giorni.

Per modificare la lunghezza della finestra di rotazione, imposta la durata della finestra. È possibile impostare la finestra di rotazione a partire da un'ora. La finestra di rotazione non deve estendersi fino alla finestra di rotazione successiva. In altre parole, per un programma di rotazione in ore, verificate che la finestra di rotazione sia inferiore o uguale al numero di ore tra le rotazioni. Per una pianificazione di rotazione in giorni, verificate che l'ora di inizio sommata alla durata della finestra sia inferiore o uguale a 24 ore.

Espressioni della frequenza

Le espressioni di frequenza di Secrets Manager hanno il seguente formato, dove *`Value`* è un numero intero positivo e *`Unit`* può esserehour, hoursday, odays:

```
rate(Value Unit)
```

Puoi ruotare un segreto anche ogni quattro ore. Il periodo di rotazione massimo è 999 giorni. Esempi:

- `rate(4 hours)` significa che il segreto viene ruotato ogni quattro ore.
- `rate(1 day)` significa che il segreto viene ruotato ogni giorno.
- `rate(10 days)` significa che il segreto viene ruotato ogni 10 giorni.

Espressioni Cron

Le espressioni cron di Secrets Manager hanno il seguente formato:

```
cron(Minutes Hours Day-of-month Month Day-of-week Year)
```

Un'espressione cron che include incrementi di ore viene ripristinata ogni giorno. Ad esempio, `cron(0 4/12 * * ? *)` significa 04:00, 16:00 e poi il giorno successivo 04:00, 16:00. I programmi di rotazione di Secrets Manager utilizzano il fuso orario UTC.

Esempio di pianificazione	Expression
Ogni otto ore a partire da mezzanotte.	`cron(0 /8 * * ? *)`
Ogni otto ore a partire dalle 08:00.	`cron(0 8/8 * * ? *)`
Ogni dieci ore, a partire dalle ore 02:00. Le finestre di rotazione inizieranno alle 02:00, alle 12:00 e alle 22:00 e poi il giorno successivo alle 02:00, alle 12:00 e alle 22:00.	`cron(0 2/10 * * ? *)`
Ogni giorno alle 10:00.	`cron(0 10 * * ? *)`
Ogni sabato alle 18:00.	`cron(0 18 ? * SAT *)`

Esempio di pianificazione	Expression
Il primo giorno di ogni mese alle 8:00.	`cron(0 8 1 * ? *)`
Ogni tre mesi, la prima domenica all'1:00.	`cron(0 1 ? 1/3 SUN#1 *)`
L'ultimo giorno di ogni mese alle 17:00.	`cron(0 17 L * ? *)`
Dal lunedì al venerdì alle 8:00.	`cron(0 8 ? * MON-FRI *)`
Il primo e il quindicesimo giorno di ogni mese alle 16:00.	`cron(0 16 1,15 * ? *)`
La prima domenica di ogni mese a mezzanotte.	`cron(0 0 ? * SUN#1 *)`
A partire da gennaio, ogni 11 mesi il primo lunedì a mezzanotte.	`cron(0 0 ? 1/11 2#1 *)`

Requisiti di espressione cron in Secrets Manager

Secrets Manager prevede alcune restrizioni su cosa può essere utilizzato per le espressioni cron. Un'espressione cron per Secrets Manager deve avere 0 nel campo dei minuti perché le finestre di rotazione di Secrets Manager iniziano ogni ora. Deve avere * nel campo dell'anno, perché Secrets Manager non supporta programmi di rotazione distanti più di un anno. Nella tabella seguente sono riportate le opzioni che puoi utilizzare.

Campi	Valori	Caratteri jolly
Minuti	Deve essere 0	Nessuno
Ore	0-23	Usa / (barra in avanti) per specificare gli incrementi. Ad esempio, `2/10` significa ogni 10 ore a partire dalle 02:00. Puoi ruotare un segreto anche ogni quattro ore.
D ay-of-month	1-31	Usa , (virgola) per includere valori aggiuntivi. Ad esempio,

Campi	Valori	Caratteri jolly
		`1,15` indica il primo e il quindicesimo giorno del mese. Usa - (trattino) per specificare un intervallo. Ad esempio, con `1-15` si intendono i giorni dal 1° al 15 del mese. Usa * (asterisco) per includere tutti i valori nel campo. Ad esempio, `*` significa ogni giorno del mese. Il carattere jolly ? (punto interrogativo) specifica un valore. Non puoi specificare i campi `Day-of-month` e `Day-of-week` nella stessa espressione cron. Se specifich i un valore in uno dei campi, devi usare un carattere ? nell'altro campo. Usa / (barra in avanti) per specificare gli incrementi. Ad esempio, `1/2` significa ogni due giorni a partire dal giorno 1 (in altre parole, il giorno 1, 3, 5 e così via). Usa L per specificare l'ultimo giorno del mese. Utilizzare ***DAY***L per specifica re l'ultimo giorno del mese.

Campi	Valori	Caratteri jolly
		Ad esempio, `SUNL` significa l'ultima domenica del mese.
Mese	1-12 o JAN-DEC	Usa , (virgola) per includere valori aggiuntivi. Ad esempio, `JAN,APR,JUL,OCT` indica gennaio, aprile, luglio e ottobre. Usa - (trattino) per specifica re un intervallo. Ad esempio, `1-3` indica dal 1° al 3° mese dell'anno. Usa * (asterisco) per includere tutti i valori nel campo. Ad esempio, `*` significa ogni mese. Usa / (barra in avanti) per specificare gli incrementi. Ad esempio, `1/3` significa ogni terzo mese, a partire dal 1° mese (in altre parole i mesi 1, 4, 7 e 10).

Campi	Valori	Caratteri jolly
D ay-of-week	1-7 o SUN-SAT	Usa # per specificare un giorno feriale all'interno del mese. Ad esempio, `TUE#3` indica il terzo martedì del mese. Usa , (virgola) per includere valori aggiuntivi. Ad esempio, `1,4` significa il primo e il quarto giorno della settimana. Usa - (trattino) per specificare un intervallo. Ad esempio, `1-4` indica i giorni dal 1° al 4° della settimana. Usa * (asterisco) per includere tutti i valori nel campo. Ad esempio, `*` significa tutti i giorni della settimana. Il carattere jolly ? (punto interrogativo) specifica un valore. Non puoi specificare i campi `Day-of-month` e `Day-of-week` nella stessa espressione cron. Se specifich i un valore in uno dei campi, devi usare un carattere ? nell'altro campo. Usa / (barra in avanti) per specificare gli incrementi. Ad esempio, `1/2` significa ogni secondo giorno della settimana, a partire dal primo

Campi	Valori	Caratteri jolly
		giorno, quindi i giorni 1, 3, 5 e 7. Usa L per specificare l'ultimo giorno della settimana.
Anno	Deve essere *	Nessuno

Ruota immediatamente un Gestione dei segreti AWS segreto

È possibile ruotare solo un segreto che ha la rotazione configurata. Per determinare se un segreto è stato configurato per la rotazione, nella console visualizza il segreto e scorri verso il basso fino alla sezione Configurazione della rotazione. Se lo Stato di rotazione è Abilitato, il segreto è configurato per la rotazione. In caso contrario, vediRotazione dei segreti .

Come ruotare immediatamente un segreto (console)

1. Apri la console Secrets Manager all'indirizzo https://console.aws.amazon.com/secretsmanager/.
2. Scegli il tuo segreto.
3. Nella pagina dei dettagli del segreto, in Configurazione rotazione, scegli Ruota immediatamente il segreto.
4. Nella finestra di dialogo Ruota segreto, scegli Ruota.

AWS CLI

Example Rotazione immediata di un segreto

L'esempio di rotate-secret seguente mostra come avviare una rotazione immediata. Il segreto deve avere già la rotazione configurata.

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

Trova segreti che non vengono ruotati

Puoi usarli AWS Config per valutare i tuoi segreti per vedere se ruotano secondo i tuoi standard. Puoi definire i requisiti interni di sicurezza e conformità per i segreti utilizzando AWS Config le regole. Quindi AWS Config puoi identificare i segreti che non sono conformi alle tue regole. È inoltre possibile tenere traccia delle modifiche ai metadati segreti, alla configurazione di rotazione, alla chiave KMS utilizzata per la crittografia segreta, alla funzione di rotazione Lambda e ai tag associati a un segreto.

Se disponi di più Account AWS segreti Regioni AWS nella tua organizzazione, puoi aggregare tali dati di configurazione e conformità. Per ulteriori informazioni, consulta Aggregazione di dati multiaccount e più regioni.

Per valutare se i segreti ruotano

1. Segui le istruzioni sulla valutazione delle tue risorse con AWS Config le regole e scegli una delle seguenti regole:
   - `secretsmanager-rotation-enabled-check`— Verifica se la rotazione è configurata per i segreti memorizzati in Secrets Manager.
   - `secretsmanager-scheduled-rotation-success-check`: verifica se l'ultima rotazione riuscita rientra nella frequenza di rotazione configurata. La frequenza minima per la verifica è giornaliera.
   - `secretsmanager-secret-periodic-rotation`— Controlla se i segreti sono stati ruotati entro il numero specificato di giorni.
2. Facoltativamente, configura AWS Config in modo che ti avvisi quando i segreti non sono conformi. Per ulteriori informazioni, consulta l'argomento Notifiche AWS Config inviate a un Amazon SNS.

Annulla la rotazione automatica in Secrets Manager

Se hai configurato la rotazione automatica per un segreto e desideri interromperne la rotazione, puoi annullare la rotazione.

Per annullare la rotazione automatica

1. Apri la console Secrets Manager all'indirizzo https://console.aws.amazon.com/secretsmanager/.
2. Scegli il tuo segreto.

3. Nella pagina dei dettagli segreti, in Configurazione della rotazione, scegli Modifica rotazione.
4. Nella finestra di dialogo Modifica configurazione di rotazione, disattiva Rotazione automatica, quindi scegli Salva.

   Secrets Manager conserva le informazioni di configurazione della rotazione in modo che possiate utilizzarle in futuro se deciderete di riattivare la rotazione.

Gestione dei segreti AWS segreti gestiti da altri AWS servizi

Molti AWS servizi archiviano e utilizzano segreti in Gestione dei segreti AWS. In alcuni casi, questi segreti sono segreti gestiti, pertanto il servizio che li ha creati aiuta anche a gestirli. Ad esempio, alcuni segreti gestiti includono la rotazione gestita, quindi non richiedono la configurazione della rotazione da parte dell'utente. Il servizio di gestione potrebbe implementare restrizioni che impediscono l'aggiornamento o l'eliminazione immediata dei segreti, al fine di garantire un periodo di recupero. Questa misura aiuta a prevenire interruzioni perché il servizio di gestione dipende dall'utilizzo corretto del segreto.

> **Note**
>
> I segreti gestiti possono essere creati solo dal AWS servizio che li gestisce.

I segreti gestiti utilizzano una convenzione di denominazione che include l'ID del servizio di gestione per aiutarli a identificarli.

```
Secret name: ServiceID!MySecret
Secret ARN : arn:aws:us-east-1:ServiceID!MySecret-a1b2c3
```

IDs per i servizi che gestiscono i segreti

- `appflow` – the section called "Amazon AppFlow"
- `databrew` – the section called "AWS Glue DataBrew"
- `datasync` – the section called "AWS DataSync"
- `directconnect` – the section called "Direct Connect"
- `ecs-sc` – the section called "Amazon Elastic Container Service"
- `events` – the section called "Amazon EventBridge"
- `marketplace-deployment` – the section called "Marketplace AWS"
- `opsworks-cm` – the section called "AWS OpsWorks for Chef Automate"
- `pcs` – the section called "AWS Servizio Parallel Computing"
- `rds` – the section called "Amazon RDS"
- `redshift` – the section called "Amazon Redshift"
- `sqlworkbench` – the section called "Editor di query v2 di Amazon Redshift"

Per trovare segreti gestiti da altri AWS servizi, vedi Trova segreti gestiti.

Servizi AWS che usano Gestione dei segreti AWS segreti

Ottieni informazioni su come ognuno dei seguenti Servizi AWS sistemi si integra con Secrets Manager.

- Come si AWS App Runner usa Gestione dei segreti AWS
- Come utilizza AWS App2Container Gestione dei segreti AWS
- Come si usa AWS AppConfig Gestione dei segreti AWS
- In che modo Amazon AppFlow utilizza Gestione dei segreti AWS
- Come si AWS AppSync usa Gestione dei segreti AWS
- Come utilizza Amazon Athena Gestione dei segreti AWS
- Come utilizza Amazon Aurora Gestione dei segreti AWS
- Come si usa AWS CodeBuild Gestione dei segreti AWS
- Come utilizza Amazon Data Firehose Gestione dei segreti AWS
- Come si usa AWS DataSync Gestione dei segreti AWS
- In che modo Amazon DataZone utilizza Gestione dei segreti AWS
- In che modo utilizza AWS Direct Connect Gestione dei segreti AWS
- Come si usa AWS Directory Service Gestione dei segreti AWS
- In che modo Amazon DocumentDB (con compatibilità con MongoDB) utilizza Gestione dei segreti AWS
- In che modo utilizza AWS Elastic Beanstalk Gestione dei segreti AWS
- Come utilizza Amazon Elastic Container Registry Gestione dei segreti AWS
- Amazon Elastic Container Service
- In che modo Amazon ElastiCache utilizza Gestione dei segreti AWS
- Come si usa AWS Elemental Live Gestione dei segreti AWS
- Come si usa AWS Elemental MediaConnect Gestione dei segreti AWS
- Come si AWS Elemental MediaConvert usa Gestione dei segreti AWS
- Come si usa AWS Elemental MediaLive Gestione dei segreti AWS
- Come si AWS Elemental MediaPackage usa Gestione dei segreti AWS
- Come si usa AWS Elemental MediaTailor Gestione dei segreti AWS

- In che modo Amazon EMR utilizza Secrets Manager
- In che modo Amazon EventBridge utilizza Gestione dei segreti AWS
- In che modo Amazon FSx utilizza Gestione dei segreti AWS i segreti
- In che modo utilizza AWS Glue DataBrew Gestione dei segreti AWS
- Come utilizza AWS Glue Studio Gestione dei segreti AWS
- Come si AWS IoT SiteWise usa Gestione dei segreti AWS
- Come utilizza Amazon Kendra Gestione dei segreti AWS
- In che modo viene utilizzato Amazon Kinesis Video Streams Gestione dei segreti AWS
- In che modo utilizza AWS Launch Wizard Gestione dei segreti AWS
- Come utilizza Amazon Lookout for Metrics Gestione dei segreti AWS
- Come utilizza Amazon Managed Grafana Gestione dei segreti AWS
- In che modo utilizza AWS Managed Services Gestione dei segreti AWS
- In che modo Amazon Managed Streaming per Apache Kafka utilizza Amazon Managed Streaming Gestione dei segreti AWS
- In che modo Amazon Managed Workflows per Apache Airflow utilizza Amazon Managed Workflows Gestione dei segreti AWS
- Marketplace AWS
- Come si AWS Migration Hub usa Gestione dei segreti AWS
- Come AWS Panorama utilizza Secrets Manager
- Come utilizza AWS Parallel Computing Service Gestione dei segreti AWS
- Come si AWS ParallelCluster usa Gestione dei segreti AWS
- In che modo Amazon Q utilizza Secrets Manager
- Come Amazon OpenSearch Ingestion utilizza Secrets Manager
- AWS OpsWorks for Chef Automate Come si utilizza Gestione dei segreti AWS
- Come utilizza Amazon Quick Gestione dei segreti AWS
- Come utilizza Amazon RDS Gestione dei segreti AWS
- Come utilizza Amazon Redshift Gestione dei segreti AWS
- Editor di query v2 di Amazon Redshift
- Come utilizza Amazon SageMaker AI Gestione dei segreti AWS
- Come si usa AWS Schema Conversion Tool Gestione dei segreti AWS
- Come utilizza Amazon Timestream per InfluxDB Gestione dei segreti AWS

- In che modo utilizza AWS Toolkit for JetBrains Gestione dei segreti AWS
- Come AWS Transfer Family utilizza i Gestione dei segreti AWS segreti
- Come AWS Wickr utilizza i Gestione dei segreti AWS segreti

Come si AWS App Runner usa Gestione dei segreti AWS

AWS App Runner è un AWS servizio che offre un modo rapido, semplice ed economico per eseguire la distribuzione dal codice sorgente o da un'immagine del contenitore direttamente a un'applicazione Web scalabile e sicura nel cloud. AWS Non è necessario apprendere nuove tecnologie, decidere quale servizio di elaborazione utilizzare o sapere come fornire e configurare le risorse. AWS

Con App Runner, puoi fare riferimento a segreti e configurazioni come variabili di ambiente nel tuo servizio quando crei un servizio o aggiorni la configurazione dello stesso. Per ulteriori informazioni, consulta le sezioni Referencing environment variables (Riferimenti alle variabili di ambiente) e Managing environment variables (Gestione delle variabili di ambiente) nella Guida per gli sviluppatori di AWS App Runner .

Come utilizza AWS App2Container Gestione dei segreti AWS

AWS App2Container è uno strumento a riga di comando che ti aiuta a trasferire e spostare le applicazioni eseguite nei tuoi data center locali o su macchine virtuali, in modo che vengano eseguite in contenitori gestiti da Amazon ECS, Amazon EKS o. AWS App Runner

App2Container utilizza Secrets Manager al fine di gestire le credenziali per connettere il computer del dipendente ai server dell'applicazione in modo da eseguire i comandi remoti. Per ulteriori informazioni, consulta Manage secrets for AWS App2Container nella Guida per l'utente di App2Container.AWS

Come si usa AWS AppConfig Gestione dei segreti AWS

AWS AppConfig è una funzionalità AWS Systems Manager che è possibile utilizzare per creare, gestire e distribuire rapidamente configurazioni di applicazioni. Una configurazione può contenere dati di credenziali o altre informazioni sensibili archiviate in Secrets Manager. Quando si crea un profilo di configurazione in formato libero, è possibile scegliere Secrets Manager come origine dei dati di configurazione. Per ulteriori informazioni, consulta Creazione di una configurazione e di un profilo di configurazione nella AWS AppConfig Guida per l'utente. Per informazioni su come AWS AppConfig gestisce i segreti con rotazione automatica attivata, vedere la rotazione dei tasti di Secrets Manager nella Guida per l'AWS AppConfig utente.

In che modo Amazon AppFlow utilizza Gestione dei segreti AWS

Amazon AppFlow è un servizio di integrazione completamente gestito che consente di scambiare dati in modo sicuro tra applicazioni SaaS (Software as a Service), come Salesforce, Amazon Simple Storage Service (Amazon S3) e Amazon Redshift. Servizi AWS

In Amazon AppFlow, quando configuri un'applicazione SaaS come origine o destinazione, crei una connessione. Ciò include le informazioni necessarie per la connessione alle applicazioni SaaS, come i token di autenticazione, i nomi utente e le password. Amazon AppFlow archivia i dati di connessione in un segreto gestito da Secrets Manager con il prefisso`appflow`. Il costo di archiviazione del segreto è incluso nell'addebito per Amazon AppFlow. Per ulteriori informazioni, consulta la sezione Protezione dei dati in Amazon AppFlow nella Amazon AppFlow User Guide.

Come si AWS AppSync usa Gestione dei segreti AWS

AWS AppSync fornisce un'interfaccia GraphQL robusta e scalabile per gli sviluppatori di applicazioni per combinare dati provenienti da più fonti, tra cui Amazon DynamoDB e HTTP. AWS Lambda APIs

AWS AppSync utilizza le credenziali in un segreto di Secrets Manager per connettersi ad Amazon RDS e Aurora. Per ulteriori informazioni, consulta il Tutorial: Aurora Serverless nella Guida per gli sviluppatori di AWS AppSync .

Come utilizza Amazon Athena Gestione dei segreti AWS

Amazon Athena è un servizio interattivo di esecuzione di query che semplifica l'analisi di dati direttamente in Amazon Simple Storage Service (Amazon S3) con SQL standard.

I connettori dell'origine dei dati Amazon Athena possono utilizzare la caratteristica Athena Federated Query con i segreti di Secrets Manager per eseguire query sui dati. Per ulteriori informazioni, consulta la sezione Utilizzo di Amazon Athena Federated Query nella Guida per l'utente di Amazon Athena.

Come utilizza Amazon Aurora Gestione dei segreti AWS

Amazon Aurora è un motore di database relazionale completamente gestito compatibile con MySQL e PostgreSQL.

Per gestire le credenziali dell'utente principale per Aurora, Aurora può creare un segreto gestito per te. Viene addebitato il costo per quel segreto. Aurora gestisce anche la rotazione di queste credenziali. Per ulteriori informazioni, consulta la sezione Gestione delle password con Amazon Aurora e Gestione dei segreti AWS nella Guida per l'utente di Amazon Aurora.

Per altre credenziali Aurora, consulta. Crea segreti

Quando chiami l'API dati di Amazon RDS, puoi passare le credenziali per il database utilizzando un secreto in Secrets Manager. Per ulteriori informazioni, consulta Utilizzo di API dati per Aurora Serverless nella Guida per l'utente di Amazon Aurora.

Quando utilizzi l'editor di query di Amazon RDS per connetterti a un database, puoi archiviare le credenziali per il database in Secrets Manager. Per ulteriori informazioni, consulta Utilizzo dell'editor di query nella Guida per l'utente di Amazon RDS.

Come si usa AWS CodeBuild Gestione dei segreti AWS

AWS CodeBuild è un servizio di compilazione completamente gestito nel cloud. CodeBuild compila il codice sorgente, esegue test unitari e produce artefatti pronti per l'implementazione.

È possibile memorizzare le credenziali del Registro di sistema privato utilizzando Secrets Manager. Per ulteriori informazioni, consulta Registro privato con Gestione dei segreti AWS esempio per CodeBuild nella Guida per l'utente.AWS CodeBuild

Come utilizza Amazon Data Firehose Gestione dei segreti AWS

Puoi utilizzare Amazon Data Firehose per distribuire dati di streaming in tempo reale a varie destinazioni di streaming. Quando la destinazione richiede una credenziale o una chiave, Firehose recupera un segreto da Secrets Manager in fase di esecuzione per connettersi alla destinazione. Per ulteriori informazioni, consulta Authenticate with Gestione dei segreti AWS in Amazon Data Firehose nella Amazon Data Firehose Developer Guide.

Come si usa AWS DataSync Gestione dei segreti AWS

AWS DataSync è un servizio di trasferimento dati online che semplifica, automatizza e accelera lo spostamento dei dati tra sistemi e servizi di storage.

Alcuni dei sistemi di storage supportati da DataSync richiedono credenziali per leggere e scrivere i dati. DataSync utilizza Secrets Manager per archiviare o accedere alle credenziali di archiviazione. È possibile DataSync configurare la creazione di segreti per conto dell'utente oppure è possibile fornire un segreto personalizzato. I segreti gestiti dal servizio iniziano con il prefisso. `aws-datasync` Ti viene addebitato solo l'uso di segreti che crei al di fuori di. DataSync Vedi Fornitura delle credenziali per le posizioni di archiviazione nella Guida per l'AWS DataSync utente.

In che modo Amazon DataZone utilizza Gestione dei segreti AWS

Amazon DataZone è un servizio di gestione dei dati che ti consente di catalogare, scoprire, governare, condividere e analizzare i tuoi dati. Puoi utilizzare risorse di dati provenienti da tabelle e viste di un cluster Amazon Redshift sottoposto a scansione tramite un processo. Crawler di AWS Glue Per connetterti ad Amazon Redshift, fornisci DataZone le credenziali Amazon in un segreto di Secrets Manager. Per ulteriori informazioni, consulta Creare un'origine dati per un database Amazon Redshift utilizzando una nuova AWS Glue connessione nella Amazon DataZone User Guide.

In che modo utilizza AWS Direct Connect Gestione dei segreti AWS

Direct Connect collega la rete interna a una Direct Connect posizione tramite un cavo Ethernet standard in fibra ottica. Con questa connessione, è possibile creare interfacce virtuali direttamente al pubblico. Servizi AWS

Direct Connect memorizza un nome di chiave di associazione di connettività e una coppia di chiavi di associazione di connettività (coppia CKN/CAK) in un segreto gestito con il prefisso. `directconnect` Il costo del segreto è incluso nel costo di. Direct Connect Per aggiornare il segreto, è necessario utilizzare Direct Connect invece di Secrets Manager. Per ulteriori informazioni, consulta Associare un MACsec CKN/CAK a un LAG nella Guida per l'Direct Connect utente.

Come si usa AWS Directory Service Gestione dei segreti AWS

Directory Service offre diversi modi per utilizzare Microsoft Active Directory (AD) con altri AWS servizi. Puoi collegare un'istanza Amazon EC2 alla directory utilizzando i segreti per le credenziali. Per ulteriori informazioni, nella Guida per l'utente di Direct Connect , consulta:

- Unisci senza problemi un'istanza Linux EC2 alla tua directory AWS Microsoft AD gestita
- Seamlessly join a Linux EC2 instance to your AD Connector directory(Collegare perfettamente un'istanza EC2 Linux alla directory AD Connector)
- Seamlessly join a Linux EC2 instance to your Simple AD directory(Collegare perfettamente un'istanza EC2 alla directory Simple AD)

In che modo Amazon DocumentDB (con compatibilità con MongoDB) utilizza Gestione dei segreti AWS

Amazon DocumentDB (compatibile con MongoDB) è un servizio di database di documenti completamente gestito che supporta carichi di lavoro MongoDB. Amazon DocumentDB si integra con

Secrets Manager per gestire le password utente principali per i cluster, migliorando la sicurezza e semplificando la gestione delle credenziali.

Amazon DocumentDB genera la password, la archivia in Secrets Manager e gestisce le impostazioni segrete. Per impostazione predefinita, Amazon DocumentDB ruota il segreto ogni sette giorni, ma puoi modificare la pianificazione di rotazione se necessario. Quando crei o modifichi un cluster Amazon DocumentDB, puoi specificare che deve gestire la password dell'utente principale in Secrets Manager. Per ulteriori informazioni, consulta Gestione delle password con Amazon DocumentDB and Secrets Manager nella Amazon DocumentDB Developer Guide.

In che modo utilizza AWS Elastic Beanstalk Gestione dei segreti AWS

Con AWS Elastic Beanstalk, puoi distribuire e gestire rapidamente le applicazioni nel AWS cloud senza dover conoscere l'infrastruttura che esegue tali applicazioni. Elastic Beanstalk può avviare ambienti Docker mediante la creazione di un'immagine descritta in un Dockerfile oppure mediante l'estrazione di un'immagine Docker in remoto. Per l'autenticazione con il registro online che ospita il repository privato, Elastic Beanstalk utilizza un segreto di Secrets Manager. Per ulteriori informazioni, consulta Configurazione di Docker nella Guida per gli sviluppatori di AWS Elastic Beanstalk .

Come utilizza Amazon Elastic Container Registry Gestione dei segreti AWS

Amazon Elastic Container Registry (Amazon ECR) è AWS un servizio di registro di immagini di container gestito sicuro, scalabile e affidabile. Puoi utilizzare la CLI di Docker, il tuo client preferito, per inviare ed estrarre immagini dai tuoi repository. Devi creare una regola di cache pull-through per ogni registro upstream contenente le immagini da memorizzare nella cache del registro privato di Amazon ECR. Per i registri upstream che richiedono l'autenticazione, devi archiviare le credenziali in un segreto di Secrets Manager. Puoi creare il segreto di Secrets Manager nelle console di Amazon ECR o di Secrets Manager. Per ulteriori informazioni, consulta Creazione di una regola pull through cache nella Amazon ECR User Guide.

Amazon Elastic Container Service

Amazon Elastic Container Service (Amazon ECS) è un servizio di orchestrazione di container completamente gestito che facilita l'implementazione, la gestione e il dimensionamento delle applicazioni containerizzate. Puoi inserire dati sensibili nei tuoi container facendo riferimento ai segreti di Secrets Manager. Per ulteriori informazioni, consulta le seguenti pagine nella Guida per gli sviluppatori di Amazon Elastic Container Service:

- Tutorial: specifica di dati sensibili utilizzando segreti Secrets Manager

- Recupero programmatico dei segreti attraverso l'applicazione
- Recupero di segreti attraverso variabili di ambiente
- Recupero di segreti per la configurazione di registrazione

Amazon ECS supporta i FSx volumi Windows File Server per contenitori. Amazon ECS utilizza le credenziali archiviate in un segreto di Secrets Manager per aggiungere il dominio ad Active Directory e collegare il file system FSx per Windows File Server. Per ulteriori informazioni, consulta Tutorial: Using FSx for Windows File Server file system con Amazon ECS e FSx per volumi Windows File Server nella Amazon Elastic Container Service Developer Guide.

È possibile fare riferimento alle immagini dei contenitori in registri privati AWS che non richiedono l'autenticazione utilizzando un segreto di Secrets Manager con le credenziali di registro. Per ulteriori informazioni, consulta Autenticazione di registri privati per i processi nella Guida per sviluppatori di Amazon Elastic Container Service.

Quando usi Amazon ECS Service Connect, Amazon ECS utilizza i segreti gestiti da Secrets Manager per archiviare i certificati AWS Autorità di certificazione privata TLS. Il costo di archiviazione del segreto è incluso nei costi di Amazon ECS. Per aggiornare il segreto, devi utilizzare Amazon ECS anziché Secrets Manager. Per ulteriori informazioni, consulta TLS with Service Connect nella Amazon Elastic Container Service Developer Guide.

In che modo Amazon ElastiCache utilizza Gestione dei segreti AWS

ElastiCache È possibile utilizzare una funzionalità chiamata Role-Based Access Control (RBAC) per proteggere il cluster. È possibile archiviare queste credenziali in Secrets Manager. Secrets Manager fornisce un modello di rotazione per questo tipo di segreto. Per ulteriori informazioni, consulta Rotazione automatica delle password per gli utenti nella Amazon ElastiCache User Guide.

Come si usa AWS Elemental Live Gestione dei segreti AWS

AWS Elemental Live è un servizio video in tempo reale che consente di creare output live per la trasmissione e la distribuzione in streaming.

AWS Elemental Live utilizza un ARN segreto per ottenere un segreto che contiene una chiave di crittografia da Secrets Manager. Elemental Live utilizza la chiave di crittografia encrypt/decrypt del video. Per ulteriori informazioni, consulta Come MediaConnect funziona la consegna da AWS Elemental Live a in fase di esecuzione nella Guida per l'utente di Elemental Live.

Come si usa AWS Elemental MediaConnect Gestione dei segreti AWS

AWS Elemental MediaConnect è un servizio che consente alle emittenti e ad altri fornitori di video premium di importare in modo affidabile video in diretta Cloud AWS e di distribuirli su più destinazioni all'interno o all'esterno di. Cloud AWS

È possibile utilizzare la crittografia a chiave statica per proteggere le origini, gli output e i diritti e archiviare la chiave di crittografia in Gestione dei segreti AWS. Per ulteriori informazioni, consulta la sezione Crittografia a chiave statica AWS Elemental MediaConnect nella Guida per l'utente.AWS Elemental MediaConnect

Come si AWS Elemental MediaConvert usa Gestione dei segreti AWS

AWS Elemental MediaConvert è un servizio di elaborazione video basato su file che fornisce un'elaborazione video scalabile per proprietari di contenuti e distributori con librerie multimediali di qualsiasi dimensione. Per MediaConvert codificare le filigrane Kantar, usi Secrets Manager per memorizzare le tue credenziali Kantar. Per ulteriori informazioni, consulta Uso di Kantar per la filigrana audio nelle uscite nella Guida per l'utente. AWS Elemental MediaConvert AWS Elemental MediaConvert

Come si usa AWS Elemental MediaLive Gestione dei segreti AWS

AWS Elemental MediaLive è un servizio video in tempo reale che consente di creare output live per la trasmissione e la distribuzione in streaming. Se l'organizzazione utilizza AWS Elemental Link dispositivi con AWS Elemental MediaLive o AWS Elemental MediaConnect, è necessario distribuire il dispositivo e configurarlo. Per ulteriori informazioni, consulta Configurazione MediaLive come entità attendibile nella Guida per l'MediaLive utente.

Come si AWS Elemental MediaPackage usa Gestione dei segreti AWS

AWS Elemental MediaPackage è un servizio di creazione e creazione di pacchetti just-in-time video che viene eseguito in. Cloud AWS Con MediaPackage, puoi distribuire flussi video altamente sicuri, scalabili e affidabili a un'ampia varietà di dispositivi di riproduzione e reti di distribuzione dei contenuti (). CDNs Per ulteriori informazioni, consulta l'accesso a Secrets Manager per l'autorizzazione CDN nella Guida per l'AWS Elemental MediaPackage utente.

Come si usa AWS Elemental MediaTailor Gestione dei segreti AWS

AWS Elemental MediaTailor è un servizio scalabile di inserimento di annunci e assemblaggio di canali che viene eseguito in. Cloud AWS

MediaTailor supporta l'autenticazione dei token di accesso di Secrets Manager alle posizioni di origine. Con l'autenticazione con token di accesso Secrets Manager, MediaTailor utilizza un segreto di Secrets Manager per autenticare le richieste all'origine. Per ulteriori informazioni, vedere Configurazione dell'autenticazione con token di Gestione dei segreti AWS accesso nella Guida per l'AWS Elemental MediaTailor utente.

In che modo Amazon EMR utilizza Secrets Manager

Amazon EMR è una piattaforma che semplifica l'esecuzione di framework di big data, come Apache Hadoop e Apache Spark, per elaborare e analizzare grandi quantità di dati. AWS Utilizzando questi framework e i relativi progetti open-source, come Apache Hive e Apache Pig, sarà possibile elaborare i dati per i carichi di lavoro di analisi e business intelligence. Puoi anche utilizzare Amazon EMR per trasformare e spostare grandi quantità di dati da e verso altri archivi di AWS dati e database, come Amazon S3 e Amazon DynamoDB.

In che modo Amazon EMR in esecuzione su Amazon EC2 utilizza Secrets Manager

Quando si crea un cluster in Amazon EMR, è possibile fornire i dati di configurazione dell'applicazione al cluster utilizzando un secreto in Secrets Manager. Per ulteriori informazioni, consulta Archiviazione dei dati sensibili di configurazione in Secrets Manager nella Guida alla gestione di Amazon EMR.

Inoltre, quando si crea un notebook EMR è possibile archiviare le credenziali del registro privato basato su Git utilizzando Secrets Manager. Per ulteriori informazioni, consulta la sezione Aggiunta di un repository basato su Git ad Amazon EMR nella Guida alla gestione di Amazon EMR.

In che modo EMR serverless utilizza Secrets Manager

EMR serverless fornisce un ambiente di runtime serverless per semplificare il funzionamento delle applicazioni di analisi in modo da non dover configurare, ottimizzare, proteggere o gestire i cluster.

È possibile archiviare i dati Gestione dei segreti AWS e quindi utilizzare l'ID segreto nelle configurazioni EMR Serverless. In questo modo, non si trasmettono dati di configurazione sensibili in testo semplice per poi esporli a fonti esterne. APIs

Per ulteriori informazioni, consulta Secrets Manager per la protezione dei dati con EMR serverless nella Guida per l'utente di Amazon EMR serverless.

In che modo Amazon EventBridge utilizza Gestione dei segreti AWS

Amazon EventBridge è un servizio di bus eventi senza server che puoi utilizzare per connettere le tue applicazioni con dati provenienti da una varietà di fonti.

Quando crei una destinazione EventBridge API Amazon, EventBridge memorizza la relativa connessione in un segreto gestito da Secrets Manager con il prefisso`events`. Il costo di archiviazione del segreto è incluso nell'addebito per l'utilizzo di una destinazione API. Per aggiornare il segreto, è necessario utilizzare EventBridge invece di Secrets Manager. Per ulteriori informazioni, consulta le destinazioni API nella Amazon EventBridge User Guide.

In che modo Amazon FSx utilizza Gestione dei segreti AWS i segreti

Amazon FSx for Windows File Server fornisce file server Microsoft Windows completamente gestiti, supportati da un file system Windows completamente nativo. Quando crei o gestisci condivisioni di file, puoi passare le credenziali di un file Gestione dei segreti AWS segreto. Per ulteriori informazioni, consulta Condivisioni di file e Migrazione delle configurazioni di condivisione di file su Amazon FSx nella Guida per l'utente di Amazon FSx per Windows File Server.

In che modo utilizza AWS Glue DataBrew Gestione dei segreti AWS

AWS Glue DataBrew è uno strumento di preparazione visiva dei dati che è possibile utilizzare per pulire e normalizzare i dati senza scrivere alcun codice. Nel DataBrew, una serie di passaggi di trasformazione dei dati viene chiamata ricetta. AWS Glue DataBrew fornisce le `DETERMINISTIC_DECRYPT`istruzioni e `CRYPTOGRAPHIC_HASH`le istruzioni per eseguire trasformazioni sulle informazioni di identificazione personale (PII) in un set di dati, che utilizzano una chiave di crittografia archiviata in un segreto di Secrets Manager. `DETERMINISTIC_ENCRYPT` Se si utilizza il segreto DataBrew predefinito per archiviare la chiave di crittografia, DataBrew crea un segreto gestito con il prefisso. `databrew` Il costo di archiviazione del segreto è incluso nel costo di utilizzo DataBrew. Se si crea un nuovo segreto per memorizzare la chiave di crittografia, DataBrew crea un segreto con il prefisso`AwsGlueDataBrew`. Viene addebitato il costo per quel segreto.

Come utilizza AWS Glue Studio Gestione dei segreti AWS

AWS Glue Studio è un'interfaccia grafica che semplifica la creazione, l'esecuzione e il monitoraggio dei lavori di estrazione, trasformazione e caricamento (ETL). AWS Glue Puoi utilizzare Amazon

OpenSearch Service come archivio dati per i tuoi lavori di estrazione, trasformazione e caricamento (ETL) configurando Elasticsearch Spark Connector in. AWS Glue Studio Per connetterti al OpenSearch cluster, puoi usare un segreto in Secrets Manager. Per ulteriori informazioni, consulta Tutorial: utilizzo del connettore AWS Glue per Elasticsearch nella Guida per glisviluppatori di AWS Glue .

Come si AWS IoT SiteWise usa Gestione dei segreti AWS

AWS IoT SiteWise è un servizio gestito che consente di raccogliere, modellare, analizzare e visualizzare dati provenienti da apparecchiature industriali su larga scala. È possibile utilizzare la AWS IoT SiteWise console per creare un gateway. Quindi aggiungi le origini dati, i server locali o le apparecchiature industriali collegate ai gateway. Se l'origine richiede l'autenticazione, utilizza un segreto per eseguirla. Per ulteriori informazioni, consulta la sezione Configuring data source authentication (Configurazione dell'autenticazione delle origini dei dati) nella Guida per l'utente di AWS IoT SiteWise .

Come utilizza Amazon Kendra Gestione dei segreti AWS

Amazon Kendra è un servizio di ricerca estremamente accurato e intelligente che consente agli utenti di cercare dati non strutturati e strutturati utilizzando l'elaborazione del linguaggio naturale e gli algoritmi di ricerca avanzata.

Puoi indicizzare i documenti archiviati in un database specificando un segreto che contiene le credenziali per il database. Per ulteriori informazioni, consulta Utilizzo di un'origine dei dati di database nella Guida per l'utente di Amazon Kendra.

In che modo viene utilizzato Amazon Kinesis Video Streams Gestione dei segreti AWS

Puoi utilizzare il flusso di video Amazon Kinesis per connetterti alle videocamere IP presso la sede del cliente, registrare e archiviare localmente i video delle telecamere e trasmettere video sul cloud per l'archiviazione a lungo termine, la riproduzione e l'elaborazione analitica. Per registrare e caricare file multimediali da videocamere IP, è necessario implementare Edge Agent del flusso di video Kinesis su AWS IoT Greengrass. Le credenziali necessarie per accedere ai file multimediali trasmessi alla videocamera vengono archiviate in un segreto di Secrets Manager. Per ulteriori informazioni, consulta Implementare Edge Agent del flusso di video Amazon Kinesis in AWS IoT Greengrass nella Guida per gli sviluppatori di flusso di video Amazon Kinesis.

In che modo utilizza AWS Launch Wizard Gestione dei segreti AWS

AWS Launch Wizard per Active Directory è un servizio che applica le best practice Cloud AWS applicative per guidare l'utente nella configurazione di una nuova infrastruttura Active Directory o nell'aggiunta di controller di dominio a un'infrastruttura esistente, in locale Cloud AWS o in locale.

AWS Launch Wizard richiede l'aggiunta delle credenziali di amministratore di dominio a Secrets Manager per aggiungere i controller di dominio ad Active Directory. Per ulteriori informazioni, consulta la sezione Configurazione di AWS Launch Wizard per Active Directory nella Guida per l'utente di AWS Launch Wizard .

Come utilizza Amazon Lookout for Metrics Gestione dei segreti AWS

Amazon Lookout for Metrics è un servizio che individua le anomalie nei dati, ne determina le cause principali e consente di intervenire rapidamente. Puoi utilizzare Amazon Redshift o Amazon RDS come origine dei dati per un rilevatore Lookout for Metrics. Per configurare l'origine dei dati, utilizza un segreto che contiene la password del database. Per ulteriori informazioni, consulta le sezioni Using Amazon RDS with Lookout for Metrics (Utilizzo di Amazon RDS con Lookout for Metrics) e Using Amazon Redshift with Lookout for Metrics (Utilizzo di Amazon Redshift con Lookout for Metrics) nella Guida per gli sviluppatori di Amazon Lookout for Metrics.

Come utilizza Amazon Managed Grafana Gestione dei segreti AWS

Grafana gestito da Amazon è un servizio di visualizzazione dei dati completamente gestito e sicuro che puoi utilizzare per interrogare, correlare e visualizzare istantaneamente parametri operativi, log e tracce da più origini. Quando utilizzi Amazon Redshift come fonte di dati, puoi fornire le credenziali di Amazon Redshift utilizzando un segreto. Gestione dei segreti AWS Per ulteriori informazioni, consulta la sezione Configurazione di Amazon Redshift nella Guida per l'utente di Grafana gestito da Amazon.

In che modo utilizza AWS Managed Services Gestione dei segreti AWS

AWS Managed Services è un servizio aziendale che fornisce la gestione continua dell' AWS infrastruttura. La modalità AMS Self-Service Provisioning (SSP) fornisce l'accesso completo alle funzionalità native Servizio AWS e API negli account gestiti AMS. Per informazioni su come richiedere l'accesso a Secrets Manager in AMS, consulta Gestione dei segreti AWS (self-service provisioning AMS) nella Guida avanzata per l'utente di AMS.

In che modo Amazon Managed Streaming per Apache Kafka utilizza Amazon Managed Streaming Gestione dei segreti AWS

Amazon Managed Streaming for Apache Kafka (Amazon MSK) è un servizio completamente gestito che consente di costruire ed eseguire applicazioni che utilizzano Apache Kafka per elaborare i dati in streaming. Puoi controllare l'accesso ai cluster Amazon MSK utilizzando i nomi utente e le password archiviati e protetti con Gestione dei segreti AWS. Per ulteriori informazioni, consulta Autenticazione nome utente e password con Gestione dei segreti AWS nella Guida per gli sviluppatori di Amazon Managed Streaming for Apache Kafka.

In che modo Amazon Managed Workflows per Apache Airflow utilizza Amazon Managed Workflows Gestione dei segreti AWS

Amazon Managed Workflows for Apache Airflow è un servizio di orchestrazione gestito per Apache Airflow che semplifica la configurazione e la gestione di pipeline di dati nel cloud su end-to-end larga scala.

Puoi configurare una connessione Apache Airflow utilizzando un segreto di Secrets Manager. Per ulteriori informazioni, consulta Configurazione di una connessione Apache Airflow utilizzando un segreto di Secrets Manager e Utilizzo di una chiave segreta per una variabile Apache Airflow nella Gestione dei segreti AWS Guida per l'utente di Amazon Managed Workflows for Apache Airflow.

Marketplace AWS

Quando usi Marketplace AWS Quick Launch, Marketplace AWS distribuisce il software insieme alla chiave di licenza. Marketplace AWS memorizza la chiave di licenza nel tuo account come segreto gestito da Secrets Manager. Il costo di archiviazione del segreto è incluso nei costi di Marketplace AWS. Per aggiornare il segreto, è necessario utilizzare Marketplace AWS invece di Secrets Manager. Per ulteriori informazioni, consulta Configura Quick Launch nella Guida per i rivenditori Marketplace AWS .

Come si AWS Migration Hub usa Gestione dei segreti AWS

AWS Migration Hub fornisce un'unica posizione per tenere traccia delle attività di migrazione su più AWS strumenti e soluzioni partner.

AWS Migration Hub Orchestrator semplifica e automatizza la migrazione di server e applicazioni aziendali verso. AWS Migration Hub Orchestrator utilizza un segreto per le informazioni di

connessione al server di origine. Per ulteriori informazioni, nella Guida per l'utente di AWS Migration Hub Orchestrator, consulta:

- Migra NetWeaver le applicazioni SAP a AWS
- Rehost applications on Amazon EC2 (Rehosting delle applicazioni su Amazon EC2)

Migration Hub Strategy Recommendations offre consigli sulla strategia di migrazione e modernizzazione relative ai percorsi di trasformazione validi per le applicazioni. Strategy Recommendations può analizzare i database di SQL Server, utilizzando un segreto per le informazioni di connessione. Per ulteriori informazioni, consulta Strategy Recommendations database analysis (Analisi del database di Strategy Recommendations).

Come AWS Panorama utilizza Secrets Manager

AWS Panorama è un servizio che porta la visione artificiale nella rete di telecamere locali. Viene utilizzato AWS Panorama per registrare un dispositivo, aggiornarne il software e distribuirvi applicazioni. Quando registri uno stream video come origini dati per la tua applicazione, se lo stream è protetto da password, AWS Panorama archivia le credenziali in un segreto di Secrets Manager. Per ulteriori informazioni, consulta Gestione dei stream delle videocamere in AWS Panorama nella Guida per sviluppatori di AWS Panorama .

Come utilizza AWS Parallel Computing Service Gestione dei segreti AWS

AWS Parallel Computing Service (AWS PCS) è un servizio gestito che semplifica l'esecuzione e la scalabilità di carichi di lavoro di calcolo ad alte prestazioni (HPC) e di apprendimento automatico distribuito. AWS

Per connettersi al cluster job scheduler, AWS PCS crea un managed secret con il prefisso `pcs` per memorizzare la chiave dello scheduler. Il costo di archiviazione del segreto è incluso nel costo del PCS. AWS AWS PCS elimina automaticamente il segreto quando si elimina il cluster AWS PCS. Per ulteriori informazioni, consulta Lavorare con i segreti del cluster in AWS PCS nella Guida per l'utente AWS PCS.

> **Important**
>
> Non modificare o eliminare i segreti del cluster AWS PCS.

Come si AWS ParallelCluster usa Gestione dei segreti AWS

AWS ParallelCluster è uno strumento di gestione dei cluster open source che è possibile utilizzare per distribuire e gestire cluster HPC (High Performance Computing) in. Cloud AWSÈ possibile creare un ambiente multiutente che includa un AWS ParallelCluster ambiente integrato con un Microsoft AD AWS gestito (Active Directory). AWS ParallelCluster Utilizza un segreto di Secrets Manager per convalidare gli accessi ad Active Directory. Per ulteriori informazioni, consulta la sezione Integrazione di Active Directory nella Guida per l'utente di AWS ParallelCluster .

In che modo Amazon Q utilizza Secrets Manager

Per autenticare Amazon Q per accedere alla tua fonte di dati, fornisci le credenziali di accesso alla fonte dati ad Amazon Q utilizzando un segreto di Secrets Manager. In caso di utilizzo della console, è possibile scegliere se creare un nuovo segreto o utilizzarne uno esistente. Per ulteriori informazioni, consulta Concepts — Authentication nella Amazon Q Developer Guide.

Come Amazon OpenSearch Ingestion utilizza Secrets Manager

Amazon OpenSearch Ingestionè un raccoglitore di dati senza server completamente gestito che trasmette log, metriche e dati di tracciamento in tempo reale a domini e raccolte di Amazon OpenSearch Service. OpenSearch Serverless Puoi utilizzare le OpenSearch Ingestion pipeline con Secrets Manager per gestire in modo sicuro le tue credenziali. Per ulteriori informazioni, consulta:

- Utilizzando una pipeline con OpenSearch IngestionAtlassian Services
- Utilizzo di una OpenSearch Ingestion pipeline con Amazon DocumentDB
- Utilizzo di una pipeline con OpenSearch IngestionConfluent Cloud Kafka
- Utilizzo di una OpenSearch Ingestion pipeline con Kafka
- Migrazione dei dati da cluster OpenSearch autogestiti utilizzando Amazon OpenSearch Ingestion

AWS OpsWorks for Chef Automate Come si utilizza Gestione dei segreti AWS

OpsWorks è un servizio di gestione della configurazione che consente di configurare e gestire le applicazioni in un'azienda cloud utilizzando OpsWorks Puppet Enterprise o AWS OpsWorks for Chef Automate.

Quando si crea un nuovo server in AWS OpsWorks CM, OpsWorks CM archivia le informazioni relative al server in un segreto gestito da Secrets Manager con il prefisso`opsworks-cm`. Il costo del segreto è incluso nell'addebito per OpsWorks. Per ulteriori informazioni, consulta Integrazione con Gestione dei segreti AWS nella Guida per l'utente di OpsWorks .

Come utilizza Amazon Quick Gestione dei segreti AWS

Amazon Quick è un servizio di business intelligence (BI) su scala cloud che puoi utilizzare per analisi, visualizzazione dei dati e reportistica. Puoi utilizzare una varietà di fonti di dati in Quick. Se si memorizzano le credenziali del database nei segreti di Secrets Manager, Quick può utilizzare tali segreti per connettersi ai database. Per ulteriori informazioni, consulta Usare Gestione dei segreti AWS segreti al posto delle credenziali del database in Amazon Quick nella Amazon Quick User Guide.

Come utilizza Amazon RDS Gestione dei segreti AWS

Amazon Relational Database Service (Amazon RDS) è un servizio Web che semplifica la configurazione, l'uso e il dimensionamento di un database relazionale in Cloud AWS.

Per gestire le credenziali utente principale per Amazon Relational Database Service (Amazon RDS), incluso Aurora, Amazon RDS può creare un segreto gestito per te. Viene addebitato il costo per quel segreto. Amazon RDS gestisce anche la rotazione per queste credenziali. Per ulteriori informazioni, consulta la sezione Gestione delle password con Amazon RDS e Gestione dei segreti AWS nella Guida per l'utente di Amazon RDS.

Per altre credenziali Amazon RDS, consulta Crea segreti.

Quando utilizzi l'editor di query di Amazon RDS per connetterti a un database, puoi archiviare le credenziali per il database in Secrets Manager. Per ulteriori informazioni, consulta Utilizzo dell'editor di query nella Guida per l'utente di Amazon RDS.

Come utilizza Amazon Redshift Gestione dei segreti AWS

Amazon Redshift è un servizio di data warehouse nel cloud in scala petabyte interamente gestito.

Per gestire le credenziali di amministratore per Amazon Redshift, Amazon Redshift può creare un segreto gestito per te. Viene addebitato il costo per quel segreto. Amazon Redshift gestisce anche la rotazione di queste credenziali. Per ulteriori informazioni, consulta Gestione delle password di amministratore Amazon Redshift utilizzando Gestione dei segreti AWS nella Guida alla gestione di Amazon Redshift.

Per altre credenziali Amazon Redshift, consulta Crea segreti.

Quando chiami l'API dati di Amazon Redshift, puoi passare le credenziali per il cluster utilizzando un secreto in Secrets Manager. Per ulteriori informazioni, consulta Uso dell'API dati di Amazon Redshift.

Quando utilizzi l'editor di query di Amazon Redshift per connetterti a un database, Amazon Redshift può archiviare le credenziali in un segreto di Secrets Manager con il prefisso `redshiftqueryeditor`. Viene addebitato il costo per quel segreto. Per ulteriori informazioni, consulta Esecuzione di query su un database con l'editor di query nella Guida alla gestione di Amazon Redshift.

Per l'editor di query v2, consulta the section called "Editor di query v2 di Amazon Redshift".

Editor di query v2 di Amazon Redshift

L'editor di query v2 di Amazon Redshift è un'applicazione client SQL basata sul Web che puoi utilizzare per creare ed eseguire query sul data warehouse Amazon Redshift. Quando utilizzi l'editor di query Amazon Redshift v2 per connetterti a un database, Amazon Redshift può archiviare le tue credenziali in un segreto gestito da Secrets Manager con il prefisso. `sqlworkbench` Il costo di archiviazione del segreto è incluso nell'addebito per Amazon Redshift. Per aggiornare il segreto, è necessario utilizzare Amazon Redshift piuttosto che Secrets Manager. Per informazioni, consulta Utilizzo dell'editor di query v2 nella Guida alla gestione di Amazon Redshift.

Per l'editor di query precedente, consulta the section called "Amazon Redshift".

Come utilizza Amazon SageMaker AI Gestione dei segreti AWS

SageMaker L'intelligenza artificiale è un servizio di machine learning completamente gestito. Con l' SageMaker intelligenza artificiale, i data scientist e gli sviluppatori possono creare e addestrare modelli di machine learning in modo rapido e semplice e quindi implementarli direttamente in un ambiente ospitato pronto per la produzione. Fornisce un'istanza del notebook di scrittura Jupyter che consente di accedere facilmente alle tue origini dati per l'esplorazione e l'analisi, in modo da non dover gestire server.

Puoi associare i repository Git all'istanza notebook Jupyter per salvare i notebook in un ambiente di controllo dell'origine che persiste anche se l'istanza viene arrestata o eliminata. È possibile gestire le credenziali dei repository privati utilizzando Secrets Manager. Per ulteriori informazioni, consulta Associare repository Git a Amazon SageMaker Notebook Instances nella Amazon SageMaker AI Developer Guide.

Per importare i dati da Databricks, Data Wrangler archivia l'URL JDBC in Secrets Manager. Per ulteriori informazioni, consulta Import data from Databricks (JDBC) (Importazione dei dati da Databricks (JDBC)).

Per importare dati da Snowflake, Data Wrangler archivia le credenziali in un segreto di Secrets Manager. Per ulteriori informazioni, consulta Import data from Snowflake (Importazione dei dati da Snowflake).

Come si usa AWS Schema Conversion Tool Gestione dei segreti AWS

È possibile utilizzare AWS Schema Conversion Tool (AWS SCT) per convertire lo schema del database esistente da un motore di database a un altro. Puoi convertire lo schema OLTP relazionale o lo schema del data warehouse. Lo schema convertito è adatto per Amazon Relational Database Service (Amazon RDS) MySQL, MariaDB, Oracle, SQL Server, database PostgreSQL, cluster Amazon Aurora DB o cluster Amazon Redshift. Lo schema convertito può essere utilizzato anche con un database su un'istanza Amazon Elastic Compute Cloud o archiviato come dati su un bucket S3.

Quando converti uno schema di database, AWS SCT puoi utilizzare le credenziali del database in cui memorizzi. Gestione dei segreti AWS Per ulteriori informazioni, vedere Utilizzo Gestione dei segreti AWS nell'interfaccia AWS SCT utente nella Guida per l'AWS Schema Conversion Tool utente.

Come utilizza Amazon Timestream per InfluxDB Gestione dei segreti AWS

Timestream for InfluxDB è un motore di database gestito di serie temporali che semplifica l'esecuzione di database InfluxDB per applicazioni di serie temporali in tempo reale utilizzando codice open source. AWS APIs Con Timestream for InfluxDB, puoi configurare, gestire e scalare carichi di lavoro di serie temporali in grado di rispondere alle query con un tempo di risposta alle query di una sola cifra di millisecondi.

Quando crei un database Timestream per InfluxDB, Timestream crea automaticamente un segreto per archiviare le credenziali di amministratore. Per ulteriori informazioni, consulta How Timestream for InfluxDB utilizza i segreti nella Timestream Developer Guide.

In che modo utilizza AWS Toolkit for JetBrains Gestione dei segreti AWS

AWS Toolkit for JetBrains È un plugin open source per gli ambienti di sviluppo integrati (IDEs) di JetBrains. Questo kit di strumenti consente agli sviluppatori di sviluppare, eseguire il debug e implementare applicazioni serverless che utilizzano AWS. Quando ti connetti a un cluster Amazon Redshift utilizzando il kit di strumenti, puoi autenticarti utilizzando un segreto di Secrets Manager.

Per ulteriori informazioni, consulta la sezione Accessing Amazon Redshift clusters (Accesso a cluster Amazon Redshift) nella Guida per l'utente di AWS Toolkit for JetBrains .

Come AWS Transfer Family utilizza i Gestione dei segreti AWS segreti

AWS Transfer Family è un servizio di trasferimento sicuro che consente di trasferire file da e verso i servizi di AWS archiviazione.

Transfer Family ora supporta l'utilizzo dell'autenticazione di base per i server che utilizzano il protocollo Applicability Statement 2 (AS2). Puoi creare un nuovo segreto di Secrets Manager o scegliere un segreto esistente per le tue credenziali. Per ulteriori informazioni, consulta Autenticazione di base per i AS2 connettori nella Guida per l'AWS Transfer Family utente.

Per autenticare gli utenti di Transfer Family, puoi utilizzarli Gestione dei segreti AWS come provider di identità. Per ulteriori informazioni, consulta Working with Custom Identity Provider nella Guida per l'AWS Transfer Family utente e l'articolo del blog Enable password authentication for AWS Transfer Family use Gestione dei segreti AWS.

È possibile utilizzare la decrittografia Pretty Good Privacy (PGP) con i file che Transfer Family elabora con i flussi di lavoro. Per utilizzare la decrittografia in una fase del flusso di lavoro, devi fornire una chiave PGP che gestisci in Secrets Manager. Per ulteriori informazioni, consulta Generare e gestire chiavi PGP nella Guida per l'utente di AWS Transfer Family .

Come AWS Wickr utilizza i Gestione dei segreti AWS segreti

AWS Wickr è un servizio end-to-end crittografato che aiuta le organizzazioni e le agenzie governative a comunicare in modo sicuro tramite one-to-one messaggistica di gruppo, chiamate vocali e video, condivisione di file, condivisione dello schermo e altro ancora. Puoi automatizzare i flussi di lavoro utilizzando i bot di conservazione dei dati di Wickr. Se il bot avrà accesso a Servizi AWS, allora dovresti creare un segreto di Secrets Manager per memorizzare le credenziali del bot. Per ulteriori informazioni, consulta Avviare il bot di conservazione dei dati nella Guida all'AWS Wickr amministrazione.

Utilizzo di segreti esterni Gestione dei segreti AWS gestiti per gestire segreti di terze parti

I segreti esterni gestiti sono un nuovo tipo di segreto Gestione dei segreti AWS che consente di archiviare e ruotare automaticamente le credenziali dei partner di integrazione. Questa funzionalità elimina la necessità di creare e gestire AWS Lambda funzioni personalizzate per la rotazione dei segreti dei partner di integrazione. Per un elenco completo di tutti i partner coinvolti, consulta Integration Partners.

Quando crei applicazioni su di esse AWS, i tuoi carichi di lavoro spesso devono interagire con applicazioni di terze parti tramite credenziali sicure come chiavi API, OAuth token o coppie di credenziali. In precedenza, era necessario sviluppare approcci personalizzati per proteggere e gestire queste credenziali, inclusa la creazione di funzioni Lambda di rotazione complesse che fossero uniche per ogni applicazione e richiedessero una manutenzione continua.

I segreti esterni gestiti forniscono un approccio standardizzato per l'archiviazione delle credenziali di terze parti in un formato predefinito prescritto da ciascun partner. La funzionalità include la rotazione automatica abilitata (per impostazione predefinita sulla console) durante la creazione segreta, la trasparenza completa e il controllo utente per i flussi di lavoro di gestione segreti e il set completo di funzionalità offerto da Secrets Manager, tra cui gestione granulare delle autorizzazioni, osservabilità, governance, conformità, disaster recovery e controlli di monitoraggio.

Funzionalità principali

I segreti esterni gestiti offrono diverse funzionalità chiave che semplificano la gestione delle credenziali di terze parti:

- La rotazione gestita senza lambda elimina il sovraccarico legato alla creazione e alla gestione di funzioni di rotazione personalizzate. Quando crei un dispositivo esterno, la rotazione viene abilitata automaticamente senza che nel tuo account siano distribuite funzioni Lambda.
- I formati segreti predefiniti assicurano che i segreti possano essere associati correttamente al partner di integrazione e includono i metadati necessari per la rotazione. Ogni partner definisce il formato richiesto.
- L'ecosistema di partner integrato fornisce supporto a più partner attraverso un processo di onboarding standardizzato. I partner si integrano direttamente con Secrets Manager per offrire linee guida programmatiche per la creazione di segreti e le funzionalità di rotazione gestite.

- La verificabilità completa mantiene la piena trasparenza attraverso la AWS CloudTrail registrazione di tutte le attività di rotazione, gli aggiornamenti segreti dei valori e le operazioni di gestione.

Partner segreti esterni gestiti

Secrets Manager si integra nativamente con applicazioni di terze parti per ruotare i segreti detenuti dal partner. Ogni partner definisce i metadati e i campi con valori segreti necessari per ruotare i segreti.

Il valore segreto contiene i campi necessari per la connessione con il client di terze parti e vengono memorizzati durante la CreateSecretchiamata. I metadati di rotazione contengono i campi utilizzati per aggiornare il segreto durante la rotazione e utilizzati nella RotateSecretchiamata. Questi campi saranno definiti dal partner di integrazione per consentire flussi di rotazione gestiti.

Affinché la rotazione funzioni correttamente, è necessario fornire a Secrets Manager autorizzazioni specifiche per gestire il ciclo di vita segreto. Per ulteriori informazioni, consulta Sicurezza e autorizzazioni

I seguenti argomenti includono una descrizione di ciascuno dei campi di metadati necessari per ruotare il segreto e una descrizione di ciascuno dei campi richiesti nel segreto di Secrets Manager per la rotazione.

Argomenti

Partner di integrazione	Tipo di segreto
Salesforce	SalesforceClientSecret
BigID	Grande IDClient segreto
Snowflake	SnowflakeKeyPairAuthentication

Segreto del cliente Salesforce

Campi di valore segreti

Di seguito sono riportati i campi che devono essere contenuti nel segreto di Secrets Manager:

```
{
```

```
  "consumerKey": "client ID",
  "consumerSecret": "client secret",
  "baseUri": "https://domain.my.salesforce.com",
  "appId": "app ID",
  "consumerId": "consumer ID"
}
```

consumerKey

La chiave consumer, nota anche come ID client, è l'identificatore di credenziali per le credenziali 2.0. OAuth È possibile recuperare la chiave consumer direttamente dalle impostazioni di Salesforce External Client App Manager. OAuth

consumerSecret

Il segreto del consumatore, noto anche come segreto del cliente, è la password privata utilizzata con la chiave consumer per l'autenticazione utilizzando il flusso di credenziali del client OAuth 2.0. È possibile recuperare il segreto del consumatore direttamente dalle impostazioni di Salesforce External Client App Manager. OAuth

baseUri

L'URI di base è l'URL di base di Salesforce Org utilizzato per interagire con Salesforce. APIs Questo assume la forma del seguente esempio:. `https://domainName.my.salesforce.com`

appId

L'ID app è l'identificatore della tua Salesforce External Client Application (ECA). Puoi recuperarlo chiamando l'endpoint Salesforce Usage. OAuth Deve iniziare con `0x` e contenere solo caratteri alfanumerici. Questo campo si riferisce a external_client_app_identifier nella guida di rotazione di Salesforce.

ConsumerID

L'ID consumatore è l'identificatore del consumatore Salesforce External Client Application (ECA). Puoi recuperarlo chiamando l'endpoint Salesforce OAuth Credentials by App ID. Questo campo si riferisce al consumer_id nella guida di rotazione di Salesforce.

Campi di metadati segreti

Di seguito sono riportati i campi di metadati necessari per ruotare un segreto detenuto da Salesforce.

```
{
```

```
  "apiVersion": "v65.0",
  "adminSecretArn": "arn:aws:secretsmanager:us-
east-1:111122223333:secret:SalesforceClientSecret"
}
```

Versione API

La versione dell'API Salesforce è la versione dell'API della tua organizzazione Salesforce. La versione deve essere almeno v65.0. Deve essere nel formato in *vXX.X* cui si *X* trova un carattere numerico.

adminSecretArn

(Facoltativo) L'ARN del segreto di amministrazione è l'Amazon Resource Name (ARN) per il segreto che contiene le OAuth credenziali amministrative da utilizzare per ruotare questo segreto del client Salesforce. Il segreto di amministrazione deve contenere almeno un valore ConsumerKey e ConsumerSecret all'interno della struttura segreta. È un campo opzionale e, se omesso, durante la rotazione Secrets Manager utilizzerà OAuth le credenziali all'interno di questo segreto per autenticarsi con Salesforce.

Flusso di utilizzo

I clienti che archiviano Salesforce Secrets in Gestione dei segreti AWS hanno la possibilità di ruotare un segreto con le credenziali archiviate nello stesso segreto o di utilizzare le credenziali nel segreto di amministrazione per la rotazione. Puoi creare il tuo segreto utilizzando la CreateSecretchiamata con il valore segreto contenente i campi sopra menzionati e il tipo segreto as. SalesforceClientSecret Le configurazioni di rotazione possono essere impostate utilizzando una RotateSecretchiamata. Questa chiamata richiede la specificazione dei campi di metadati come nell'esempio precedente. Se opti per una rotazione utilizzando le credenziali nello stesso segreto, puoi saltare il campo. adminSecretArn Inoltre, i clienti devono fornire un ruolo ARN nella RotateSecretchiamata che concede al servizio le autorizzazioni necessarie per ruotare il segreto. Per un esempio di politica di autorizzazioni, vedi Sicurezza e autorizzazioni.

Per i clienti che scelgono di ruotare i propri segreti utilizzando un set separato di credenziali (archiviate in un Admin Secret), assicurati di creare l'Admin Secret Gestione dei segreti AWS seguendo esattamente la stessa procedura utilizzata per il tuo segreto di consumo. È necessario fornire l'ARN di questo segreto di amministrazione nei metadati di rotazione in una RotateSecretchiamata per il segreto del consumatore.

La logica di rotazione segue le indicazioni fornite da Salesforce.

Token di aggiornamento Big ID

Campi con valori segreti

Di seguito sono riportati i campi che devono essere contenuti nel segreto di Secrets Manager:

```
{
  "hostname": "Host Name",
  "refreshToken": "Refresh Token"
}
```

hostname

Questo è il nome host in cui è ospitata l'istanza BigID. Devi inserire il nome di dominio completo della tua istanza.

RefreshToken

Il token di aggiornamento utente JWT generato nella console BigID tramite Amministrazione → Gestione accessi → Seleziona utente → Genera token → Salva

Flusso di utilizzo

Puoi creare il tuo segreto utilizzando la CreateSecretchiamata con il valore segreto contenente i campi sopra menzionati e il tipo segreto come Big IDClient Secret. Le configurazioni di rotazione possono essere impostate utilizzando una RotateSecretchiamata. È inoltre necessario fornire un ruolo ARN nella RotateSecretchiamata che conceda al servizio le autorizzazioni necessarie per ruotare il segreto. Per un esempio di politica di autorizzazioni, vedi Sicurezza e autorizzazioni. Nota che il campo dei metadati di rotazione può essere lasciato vuoto per questo partner.

Coppia di chiavi Snowflake

Campi di valore segreti

Di seguito sono riportati i campi che devono essere contenuti nel segreto di Secrets Manager:

```
{
  "account": "Your Account Identifier",
  "user": "Your user name",
  "privateKey": "Your private Key",
  "publicKey": "Your public Key",
```

```
  "passphrase": "Your Passphrase"
}
```

user

Il nome utente Snowflake associato a questa autenticazione a coppia di chiavi. Questo utente deve essere configurato in Snowflake per accettare l'autenticazione tramite coppia di chiavi e la chiave pubblica deve essere assegnata al profilo di questo utente.

account

L'identificatore dell'account Snowflake utilizzato per stabilire la connessione. Questo può essere estratto dal tuo URL Snowflake (la parte precedente a .snowflakecomputing.com)

privateKey

La chiave privata RSA in formato PEM utilizzata per l'autenticazione. I BEGIN/END marker sono opzionali.

Chiave pubblica

La controparte a chiave pubblica in formato PEM corrispondente alla chiave privata. I BEGIN/END marker sono opzionali.

Passphrase

(Facoltativo) Questo campo si riferisce alla passphrase utilizzata per decrittografare la chiave privata crittografata.

Campi di metadati segreti

Di seguito sono riportati i campi di metadati per Snowflake:

```
{
  "cryptographicAlgorithm": "Your Cryptographic algorithm",
  "encryptPrivateKey": "True/False"
}
```

Algoritmo crittografico

(Facoltativo) Si riferisce all'algoritmo utilizzato per la generazione delle chiavi. È possibile scegliere tra 3 algoritmi:`RS256 | RS384 | RS512`. Questo campo è facoltativo e l'algoritmo predefinito scelto è RS256.

encryptPrivateKey

(Facoltativo) Questo campo può essere utilizzato per scegliere se crittografare la chiave privata. Per impostazione predefinita, è impostato su false. La passphrase per la crittografia viene generata casualmente.

Flusso di utilizzo

Puoi creare il tuo segreto utilizzando la CreateSecretchiamata con il valore segreto contenente i campi sopra menzionati e il tipo segreto as SnowflakeKeyPairAuthentication. Le configurazioni di rotazione possono essere impostate utilizzando una RotateSecretchiamata. Facoltativamente, puoi fornire i campi di metadati segreti in base alle tue esigenze. È inoltre necessario fornire un ruolo ARN nella RotateSecretchiamata che conceda al servizio le autorizzazioni necessarie per ruotare il segreto. Per un esempio di politica di autorizzazioni, vedi Sicurezza e autorizzazioni. Nota che il campo dei metadati di rotazione può essere lasciato vuoto per questo partner.

Sicurezza e autorizzazioni

I segreti esterni gestiti non richiedono la condivisione dei privilegi a livello di amministratore degli account delle applicazioni di terze parti con. AWS Il processo di rotazione utilizza invece le credenziali e i metadati forniti dall'utente per effettuare chiamate API autorizzate all'applicazione di terze parti per l'aggiornamento e la convalida delle credenziali.

I segreti esterni gestiti mantengono gli stessi standard di sicurezza degli altri tipi di segreti di Secrets Manager. I valori segreti vengono crittografati quando sono inattivi utilizzando le chiavi KMS e in transito tramite TLS. L'accesso ai segreti è controllato tramite politiche IAM e politiche basate sulle risorse. Quando utilizzi una chiave gestita dal cliente per crittografare il tuo segreto, dovrai aggiornare la policy IAM relativa al ruolo di rotazione e la policy di fiducia CMK per fornire le autorizzazioni necessarie per garantire una rotazione di successo.

Affinché la rotazione funzioni correttamente, è necessario fornire a Secrets Manager autorizzazioni specifiche per gestire il ciclo di vita segreto. Queste autorizzazioni possono essere limitate a singoli segreti e seguire il principio del privilegio minimo. Il ruolo di rotazione fornito viene convalidato durante la configurazione e utilizzato esclusivamente per le operazioni di rotazione.

Puoi limitare l'ingresso IP alla tua risorsa esterna autorizzando solo gli intervalli AWS IP per EC2 nella regione in cui esiste il tuo segreto. Questo elenco di intervalli IP può cambiare, quindi è necessario aggiornare periodicamente le regole di ingresso.

Gestione dei segreti AWS offre anche soluzioni single touch per creare la policy IAM con le autorizzazioni necessarie per gestire il segreto durante la creazione del segreto tramite la console Secrets Manager. Le autorizzazioni per questo ruolo sono limitate per ogni partner di integrazione in ogni regione.

Esempio di politica sulle autorizzazioni:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowRotationAccess",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "secretsmanager:UpdateSecretVersionStage"
      ],
      "Resource": "*",
      "Effect": "Allow",
      "Condition": {
        "StringEquals": {
          "secretsmanager:resource/Type": "SalesforceClientSecret"
        }
      }
    },
    {
      "Sid": "AllowPasswordGenerationAccess",
      "Action": [
        "secretsmanager:GetRandomPassword"
      ],
      "Resource": "*",
      "Effect": "Allow"
    }
  ]
}
```

Nota: l'elenco dei tipi di segreti disponibili per SecretsManager:Resource/type è disponibile in Integration Partners.

Esempio di politica di fiducia:

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "SecretsManagerPrincipalAccess",
      "Effect": "Allow",
      "Principal": {
        "Service": "secretsmanager.amazonaws.com"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "111122223333"
        },
        "ArnLike": {
          "aws:SourceArn": "arn:aws:secretsmanager:us-east-1:111122223333:secret:*"
        }
      }
    }
  ]
}
```

Monitora e risolvi i problemi relativi ai segreti esterni gestiti

I segreti esterni gestiti forniscono funzionalità di monitoraggio complete tramite AWS CloudTrail log e CloudWatch metriche Amazon. Tutte le attività di rotazione vengono registrate con informazioni dettagliate su esito positivo, negativo ed eventuali errori riscontrati durante il processo.

I problemi più comuni nel flusso di lavoro di rotazione includono una configurazione errata delle autorizzazioni dei ruoli o del valore segreto. La mancata impostazione di questi campi è il formato specificato dai partner di integrazione e può causare errori di rotazione, in quanto il servizio non sarà in grado di accedere al segreto o connettersi con il client del partner di integrazione per aggiornare il segreto. Altri problemi potrebbero essere problemi di connettività di rete, scadenza delle credenziali o disponibilità del servizio partner. Il servizio di rotazione gestita include la logica dei tentativi e la gestione degli errori per massimizzare l'affidabilità

Puoi monitorare i piani di rotazione, le percentuali di successo e le metriche delle prestazioni tramite Amazon CloudWatch. Puoi configurare allarmi personalizzati tramite Event Bridge per avvisarti di errori di rotazione o altri problemi che richiedono attenzione.

Migrazione dei segreti esistenti

Hai la possibilità di migrare i segreti dei partner esistenti verso segreti esterni gestiti. Questo può essere fatto con una UpdateSecretchiamata. È necessario aggiornare il valore segreto e i metadati come indicato nella guida. Se hai già impostato una logica di rotazione personalizzata per questi segreti, devi prima annullare la rotazione utilizzando una CancelRotateSecretchiamata.

Considerazioni e limitazioni

I segreti esterni gestiti non supportano segreti effimeri con una durata inferiore a quattro ore. Inoltre, i segreti associati ai certificati dell'infrastruttura a chiave pubblica non sono supportati.

I segreti esterni gestiti sono supportati solo per i partner che hanno aderito. Gestione dei segreti AWS Per un elenco completo, consulta Integration Partners. Non vedi il tuo partner nell'elenco? Digli a Onboard di Gestione dei segreti AWS

Se si aggiornano o si ruotano i valori segreti direttamente dal servizio client partner all'esterno del motore di rotazione di Secrets Manager, la sincronizzazione tra i sistemi potrebbe interrompersi. Sebbene Secrets Manager fornisca avvisi sulla console e prevenzione programmatica per gli aggiornamenti manuali dei valori segreti, puoi comunque modificare i valori direttamente nell'applicazione di terze parti. Per ristabilire la sincronizzazione dopo out-of-band gli aggiornamenti, è necessario aggiornare il valore segreto in modo che rifletta il segreto corretto e quindi richiamare l'RotateSecretAPI per garantire che le rotazioni continuino con successo.

Crea Gestione dei segreti AWS segreti in AWS CloudFormation

È possibile creare segreti in una CloudFormation pila utilizzando la `AWS::SecretsManager::Secret` risorsa in un CloudFormation modello, come illustrato inCreazione di un segreto.

Per creare un segreto di amministrazione per Amazon RDS o Aurora, ti consigliamo di utilizzare `ManageMasterUserPassword` in `AWS::RDS::DBCluster`. Quindi Amazon RDS crea il segreto e gestisce la rotazione per te. Per ulteriori informazioni, consulta Rotazione gestita.

Per le credenziali Amazon Redshift e Amazon DocumentDB, innanzitutto crea un segreto con una password generata da Secrets Manager e quindi utilizza un riferimento dinamico per recuperare il nome utente e la password dal segreto da utilizzare come credenziali per un nuovo database. Quindi, usa la risorsa `AWS::SecretsManager::SecretTargetAttachment` per aggiungere dettagli sul database al segreto necessari a Secrets Manager per ruotare il segreto. Infine, per attivare la rotazione automatica, utilizza la risorsa `AWS::SecretsManager::RotationSchedule` e fornisci una funzione di rotazione e una pianificazione. Fare riferimento agli esempi riportati di seguito:

- Crea un segreto con le credenziali Amazon Redshift
- Crea un segreto con le credenziali Amazon DocumentDB

Per allegare una policy delle risorse al segreto, usa la risorsa `AWS::SecretsManager::ResourcePolicy`.

Per informazioni sulla creazione di risorse con CloudFormation, consulta Impara le nozioni di base sui modelli nella Guida per l' CloudFormation utente. Puoi anche utilizzare l' AWS Cloud Development Kit (AWS CDK). Per ulteriori informazioni, consulta Gestione dei segreti AWS Libreria Construct.

Crea un Gestione dei segreti AWS segreto con CloudFormation

Questo esempio crea un segreto denominato **`CloudFormationCreatedSecret-a1b2c3d4e5f6`**. Il valore del segreto è il seguente JSON, con una password di 32 caratteri che viene generata al momento della creazione del segreto.

```
{
```

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

Questo esempio utilizza la seguente CloudFormation risorsa:

- AWS::SecretsManager::Secret

Per informazioni sulla creazione di risorse con CloudFormation, consulta Impara le nozioni di base sui modelli nella Guida per l' CloudFormation utente.

JSON

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

YAML

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

Crea un Gestione dei segreti AWS segreto con rotazione automatica e un'istanza DB Amazon RDS MySQL con CloudFormation

Per creare un segreto di amministrazione per Amazon RDS o Aurora, ti consigliamo di utilizzare `ManageMasterUserPassword`, come mostrato nell'esempio Crea un segreto Secrets Manager per una password principale in `AWS::RDS::DBCluster`. Quindi Amazon RDS crea il segreto e gestisce la rotazione per te. Per ulteriori informazioni, consulta Rotazione gestita.

Crea un cluster Gestione dei segreti AWS segreto e un cluster Amazon Redshift con CloudFormation

Per creare un segreto amministrativo per Amazon Redshift, ti consigliamo di utilizzare gli esempi su `AWS::Redshift::Cluster`e. `AWS::RedshiftServerless::Namespace`

Crea un'istanza Gestione dei segreti AWS segreta e un'istanza Amazon DocumentDB con CloudFormation

In questo esempio vengono creati un segreto e un'istanza Amazon DocumentDB che utilizzano le credenziali nel segreto come utente e password. Il segreto ha una policy basata sulle risorse collegata che definisce chi può accedere al segreto. Il modello, inoltre, crea una funzione di rotazione Lambda e configura il segreto da Modelli di funzione di rotazione affinché ruoti automaticamente tra le 8:00 e le 10:00 UTC il primo giorno di ogni mese. Come best practice di sicurezza, l'istanza si trova in un Amazon VPC.

Questo esempio utilizza le seguenti CloudFormation risorse per Secrets Manager:

- `AWS::SecretsManager::Secret`
- `AWS::SecretsManager::SecretTargetAttachment`
- `AWS::SecretsManager::RotationSchedule`

Per informazioni sulla creazione di risorse con CloudFormation, consulta Impara le nozioni di base sui modelli nella Guida per l' CloudFormation utente.

JSON

```
{
   "AWSTemplateFormatVersion":"2010-09-09",
   "Transform":"AWS::SecretsManager-2020-07-23",
   "Resources":{
      "TestVPC":{
         "Type":"AWS::EC2::VPC",
         "Properties":{
            "CidrBlock":"10.0.0.0/16",
            "EnableDnsHostnames":true,
            "EnableDnsSupport":true
         }
      },
      "TestSubnet01":{
         "Type":"AWS::EC2::Subnet",
         "Properties":{
            "CidrBlock":"10.0.96.0/19",
            "AvailabilityZone":{
               "Fn::Select":[
                  "0",
                  {
                     "Fn::GetAZs":{
                        "Ref":"AWS::Region"
                     }
                  }
               ]
            },
            "VpcId":{
               "Ref":"TestVPC"
            }
         }
      },
      "TestSubnet02":{
         "Type":"AWS::EC2::Subnet",
         "Properties":{
            "CidrBlock":"10.0.128.0/19",
            "AvailabilityZone":{
               "Fn::Select":[
                  "1",
                  {
                     "Fn::GetAZs":{
                        "Ref":"AWS::Region"
```

```
                    }
                }
            ]
        },
        "VpcId":{
            "Ref":"TestVPC"
        }
    }
},
"SecretsManagerVPCEndpoint":{
    "Type":"AWS::EC2::VPCEndpoint",
    "Properties":{
        "SubnetIds":[
            {
                "Ref":"TestSubnet01"
            },
            {
                "Ref":"TestSubnet02"
            }
        ],
        "SecurityGroupIds":[
            {
                "Fn::GetAtt":[
                    "TestVPC",
                    "DefaultSecurityGroup"
                ]
            }
        ],
        "VpcEndpointType":"Interface",
        "ServiceName":{
            "Fn::Sub":"com.amazonaws.${AWS::Region}.secretsmanager"
        },
        "PrivateDnsEnabled":true,
        "VpcId":{
            "Ref":"TestVPC"
        }
    }
},
"MyDocDBClusterRotationSecret":{
    "Type":"AWS::SecretsManager::Secret",
    "Properties":{
        "GenerateSecretString":{
            "SecretStringTemplate":"{\"username\": \"someadmin\",\"ssl\": true}",
            "GenerateStringKey":"password",
```

```
               "PasswordLength":16,
               "ExcludeCharacters":"\"@/\\"
            },
            "Tags":[
               {
                  "Key":"AppName",
                  "Value":"MyApp"
               }
            ]
         }
      },
      "MyDocDBCluster":{
         "Type":"AWS::DocDB::DBCluster",
         "Properties":{
            "DBSubnetGroupName":{
               "Ref":"MyDBSubnetGroup"
            },
            "MasterUsername":{
               "Fn::Sub":"{{resolve:secretsmanager:
${MyDocDBClusterRotationSecret}::username}}"
            },
            "MasterUserPassword":{
               "Fn::Sub":"{{resolve:secretsmanager:
${MyDocDBClusterRotationSecret}::password}}"
            },
            "VpcSecurityGroupIds":[
               {
                  "Fn::GetAtt":[
                     "TestVPC",
                     "DefaultSecurityGroup"
                  ]
               }
            ]
         }
      },
      "DocDBInstance":{
         "Type":"AWS::DocDB::DBInstance",
         "Properties":{
            "DBClusterIdentifier":{
               "Ref":"MyDocDBCluster"
            },
            "DBInstanceClass":"db.r5.large"
         }
      },
```

```
    "MyDBSubnetGroup":{
       "Type":"AWS::DocDB::DBSubnetGroup",
       "Properties":{
          "DBSubnetGroupDescription":"",
          "SubnetIds":[
             {
                "Ref":"TestSubnet01"
             },
             {
                "Ref":"TestSubnet02"
             }
          ]
       }
    },
    "SecretDocDBClusterAttachment":{
       "Type":"AWS::SecretsManager::SecretTargetAttachment",
       "Properties":{
          "SecretId":{
             "Ref":"MyDocDBClusterRotationSecret"
          },
          "TargetId":{
             "Ref":"MyDocDBCluster"
          },
          "TargetType":"AWS::DocDB::DBCluster"
       }
    },
    "MySecretRotationSchedule":{
       "Type":"AWS::SecretsManager::RotationSchedule",
       "DependsOn":"SecretDocDBClusterAttachment",
       "Properties":{
          "SecretId":{
             "Ref":"MyDocDBClusterRotationSecret"
          },
          "HostedRotationLambda":{
             "RotationType":"MongoDBSingleUser",
             "RotationLambdaName":"MongoDBSingleUser",
             "VpcSecurityGroupIds":{
                "Fn::GetAtt":[
                   "TestVPC",
                   "DefaultSecurityGroup"
                ]
             },
             "VpcSubnetIds":{
                "Fn::Join":[
```

```
                    ",",
                    [
                       {
                          "Ref":"TestSubnet01"
                       },
                       {
                          "Ref":"TestSubnet02"
                       }
                    ]
                 ]
              }
           },
           "RotationRules":{
             "Duration": "2h",
             "ScheduleExpression": "cron(0 8 1 * ? *)"
           }
        }
     }
  }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::SecretsManager-2020-07-23
Resources:
  TestVPC:
    Type: AWS::EC2::VPC
    Properties:
      CidrBlock: 10.0.0.0/16
      EnableDnsHostnames: true
      EnableDnsSupport: true
  TestSubnet01:
    Type: AWS::EC2::Subnet
    Properties:
      CidrBlock: 10.0.96.0/19
      AvailabilityZone: !Select
        - '0'
        - !GetAZs
          Ref: AWS::Region
      VpcId: !Ref TestVPC
  TestSubnet02:
    Type: AWS::EC2::Subnet
```

```
    Properties:
      CidrBlock: 10.0.128.0/19
      AvailabilityZone: !Select
        - '1'
        - !GetAZs
          Ref: AWS::Region
      VpcId: !Ref TestVPC
  SecretsManagerVPCEndpoint:
    Type: AWS::EC2::VPCEndpoint
    Properties:
      SubnetIds:
        - !Ref TestSubnet01
        - !Ref TestSubnet02
      SecurityGroupIds:
        - !GetAtt TestVPC.DefaultSecurityGroup
      VpcEndpointType: Interface
      ServiceName: !Sub com.amazonaws.${AWS::Region}.secretsmanager
      PrivateDnsEnabled: true
      VpcId: !Ref TestVPC
  MyDocDBClusterRotationSecret:
    Type: AWS::SecretsManager::Secret
    Properties:
      GenerateSecretString:
        SecretStringTemplate: '{"username": "someadmin","ssl": true}'
        GenerateStringKey: password
        PasswordLength: 16
        ExcludeCharacters: '"@/\'
      Tags:
        - Key: AppName
          Value: MyApp
  MyDocDBCluster:
    Type: AWS::DocDB::DBCluster
    Properties:
      DBSubnetGroupName: !Ref MyDBSubnetGroup
      MasterUsername: !Sub '{{resolve:secretsmanager:
${MyDocDBClusterRotationSecret}::username}}'
      MasterUserPassword: !Sub '{{resolve:secretsmanager:
${MyDocDBClusterRotationSecret}::password}}'
      VpcSecurityGroupIds:
        - !GetAtt TestVPC.DefaultSecurityGroup
  DocDBInstance:
    Type: AWS::DocDB::DBInstance
    Properties:
      DBClusterIdentifier: !Ref MyDocDBCluster
```

```
      DBInstanceClass: db.r5.large
  MyDBSubnetGroup:
    Type: AWS::DocDB::DBSubnetGroup
    Properties:
      DBSubnetGroupDescription: ''
      SubnetIds:
        - !Ref TestSubnet01
        - !Ref TestSubnet02
  SecretDocDBClusterAttachment:
    Type: AWS::SecretsManager::SecretTargetAttachment
    Properties:
      SecretId: !Ref MyDocDBClusterRotationSecret
      TargetId: !Ref MyDocDBCluster
      TargetType: AWS::DocDB::DBCluster
  MySecretRotationSchedule:
    Type: AWS::SecretsManager::RotationSchedule
    DependsOn: SecretDocDBClusterAttachment
    Properties:
      SecretId: !Ref MyDocDBClusterRotationSecret
      HostedRotationLambda:
        RotationType: MongoDBSingleUser
        RotationLambdaName: MongoDBSingleUser
        VpcSecurityGroupIds: !GetAtt TestVPC.DefaultSecurityGroup
        VpcSubnetIds: !Join
          - ','
          - - !Ref TestSubnet01
            - !Ref TestSubnet02
      RotationRules:
        Duration: 2h
        ScheduleExpression: cron(0 8 1 * ? *)
```

Come utilizza Secrets Manager AWS CloudFormation

Quando si utilizza la console per attivare la rotazione, Secrets Manager crea risorse per la rotazione. AWS CloudFormation Se crei una nuova funzione di rotazione durante quel processo, ne CloudFormation crea una `AWS::Serverless::Function`basata su quella appropriataModelli di funzione di rotazione. Quindi CloudFormation imposta il `RotationSchedule`, che imposta la funzione di rotazione e le regole di rotazione per il segreto. Puoi visualizzare la CloudFormation pila selezionando Visualizza pila nel banner dopo aver attivato la rotazione automatica.

Per informazioni sull'attivazione della rotazione automatica, consulta Rotazione dei segreti .

Crea Gestione dei segreti AWS segreti in AWS Cloud Development Kit (AWS CDK)

Per creare, gestire e recuperare segreti in un'app CDK, puoi utilizzare la Libreria dei costrutti di Gestione dei segreti AWS, che contiene costrutti `ResourcePolicy`, `RotationSchedule`, `Secret`, `SecretRotation` e `SecretTargetAttachment`.

Una buona pratica per utilizzare i segreti nelle applicazioni CDK consiste nel creare prima il segreto utilizzando la console o la CLI, quindi importare il segreto nell'applicazione CDK.

Per degli esempi, consulta:

- Creazione di un segreto
- Importazione di un segreto
- Recupero di un segreto
- Concessione del permesso di utilizzare il segreto
- Rotazione di un segreto
- Rotazione di un segreto del database
- Replica un segreto in altre regioni

Per ulteriori informazioni sulla CDK, consulta la Guida per gli sviluppatori di AWS Cloud Development Kit (AWS CDK) v2.

Monitora Gestione dei segreti AWS i segreti

AWS fornisce strumenti di monitoraggio per controllare i segreti di Secrets Manager, segnalare quando qualcosa non va e intraprendere azioni automatiche quando necessario. Puoi usare il registro per analizzare qualsiasi modifica o utilizzo imprevisto e le modifiche indesiderate possono essere annullate. È inoltre possibile impostare dei controlli automatici per l'uso inappropriato dei segreti e per qualsiasi tentativo di eliminarli.

Argomenti

Registra Gestione dei segreti AWS gli eventi con AWS CloudTrail

AWS CloudTrail registra tutte le chiamate API per Secrets Manager come eventi, incluse le chiamate dalla console Secrets Manager, oltre a diversi altri eventi per la rotazione e l'eliminazione di versioni segrete. Per un elenco delle voci di registro nei record di Secrets Manager, vedereCloudTrail voci.

È possibile utilizzare la CloudTrail console per visualizzare gli ultimi 90 giorni di eventi registrati. Per una registrazione continua degli eventi nel tuo AWS account, inclusi gli eventi per Secrets Manager, crea un percorso in modo che CloudTrail invii i file di registro a un bucket Amazon S3. Vedi Creazione di un percorso per il tuo AWS account. Puoi anche configurare la ricezione CloudTrail di file di CloudTrail registro da più Account AWS e Regioni AWS.

È possibile configurare altri AWS servizi per analizzare ulteriormente e agire in base ai dati raccolti nei CloudTrail log. Visualizza le integrazioni AWS dei servizi con CloudTrail i log. Puoi anche ricevere notifiche quando CloudTrail pubblica nuovi file di log nel tuo bucket Amazon S3. Consulta Configurazione delle notifiche di Amazon SNS per. CloudTrail

Per recuperare gli eventi di Secrets Manager dai CloudTrail registri (console)

1. Apri la CloudTrail console all'indirizzo. https://console.aws.amazon.com/cloudtrail/
2. Assicurati che la console punti alla regione in cui si sono verificati gli eventi. La console mostra solo gli eventi che si sono verificati nella regione selezionata. Scegli la regione dall'elenco a discesa nell'angolo in alto a destra della console.
3. Nel riquadro di navigazione a sinistra, seleziona Event history (Cronologia eventi).
4. Scegli Filtra i criteri and/or e l'intervallo di tempo per aiutarti a trovare l'evento che stai cercando. Esempio:
   a. Per visualizzare tutti gli eventi di Secrets Manager, per gli attributi di ricerca, scegli Origine evento. Quindi per Enter event source (Inserisci origine evento) scegli **`secretsmanager.amazonaws.com`**.
   b. Per visualizzare tutti gli eventi relativi a un segreto, per gli attributi di ricerca, scegli Nome risorsa. Quindi, per Inserisci il nome di una risorsa, inserisci il nome del segreto.
5. Per visualizzare ulteriori dettagli, scegli la freccia di espansione accanto all'evento. Per visualizzare tutte le informazioni disponibili, scegli View event (Visualizza evento).

AWS CLI

Example Recupera gli eventi di Secrets Manager dai registri CloudTrail

L'esempio di `lookup-events` seguente mostra la ricerca degli eventi di Secrets Manager.

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

AWS CloudTrail iscrizioni per Secrets Manager

Gestione dei segreti AWS scrive le voci nel AWS CloudTrail registro per tutte le operazioni di Secrets Manager e per altri eventi relativi alla rotazione e all'eliminazione. Per informazioni su come intervenire su questi eventi, consulta Abbina gli eventi di Secrets Manager con EventBridge.

Tipi di voci del file di log

- Voce del file di log per le operazioni di Secrets Manager

- Registra le voci del file di log da eliminare
- Voci di log per la replica
- Registra le voci del file di log per la rotazione

Voce del file di log per le operazioni di Secrets Manager

Gli eventi generati dalle chiamate alle operazioni di Secrets Manager hanno `"detail-type": ["AWS API Call via CloudTrail"]`.

> **Note**
>
> Prima di febbraio 2024, alcune operazioni di Secrets Manager segnalavano eventi che contenevano «ArN» anziché "arn" per l'ARN segreto. Per ulteriori informazioni, consulta AWS re:Post.

Di seguito sono riportate le CloudTrail voci generate quando l'utente o un servizio richiama le operazioni di Secrets Manager tramite API, SDK o CLI.

BatchGetSecretValue

Generato dall'operazione. BatchGetSecretValue Per ulteriori informazioni sul recupero dei segreti, consulta Ottieni segreti.

CancelRotateSecret

Generato dall'CancelRotateSecretoperazione. Per ulteriori informazioni sulla rotazione, consulta Rotazione dei segreti .

CreateSecret

Generato dall'CreateSecretoperazione. Per ulteriori informazioni sulla creazione di un segreto, consulta Gestisci i segreti.

DeleteResourcePolicy

Generato dall'DeleteResourcePolicyoperazione. Per informazioni sulle autorizzazioni, consulta the section called “Autenticazione e controllo degli accessi”.

DeleteSecret

Generato dall'DeleteSecretoperazione. Per informazioni sull'eliminazione dei segreti, consulta the section called “Eliminare un segreto”.

DescribeSecret

Generato dall'DescribeSecretoperazione.

GetRandomPassword

Generato dall'GetRandomPasswordoperazione.

GetResourcePolicy

Generato dall'GetResourcePolicyoperazione. Per informazioni sulle autorizzazioni, consulta the section called “Autenticazione e controllo degli accessi”.

GetSecretValue

Generato dalle BatchGetSecretValueoperazioni GetSecretValuee. Per ulteriori informazioni sul recupero dei segreti, consulta Ottieni segreti.

ListSecrets

Generato dall'ListSecretsoperazione. Per ulteriori informazioni sulla visualizzazione dei segreti, consulta the section called “Scopri i segreti”.

ListSecretVersionIds

Generato dall'ListSecretVersionIdsoperazione.

PutResourcePolicy

Generato dall'PutResourcePolicyoperazione. Per informazioni sulle autorizzazioni, consulta the section called “Autenticazione e controllo degli accessi”.

PutSecretValue

Generato dall'PutSecretValueoperazione. Per ulteriori informazioni sull'aggiornamento di un segreto, consulta the section called “Modificare un segreto”.

RemoveRegionsFromReplication

Generato dall'RemoveRegionsFromReplicationoperazione. Per ulteriori informazioni sulla replica di un segreto, consulta Replica in più regioni.

ReplicateSecretToRegions

Generato dall'ReplicateSecretToRegionsoperazione. Per ulteriori informazioni sulla replica di un segreto, consulta Replica in più regioni.

RestoreSecret

Generato dall'RestoreSecretoperazione. Per informazioni sul ripristino di un segreto eliminato, consulta the section called “Ripristino di un segreto”.

RotateSecret

Generato dall'RotateSecretoperazione. Per ulteriori informazioni sulla rotazione, consulta Rotazione dei segreti .

StopReplicationToReplica

Generato dall'StopReplicationToReplicaoperazione. Per ulteriori informazioni sulla replica di un segreto, consulta Replica in più regioni.

TagResource

Generato dall'TagResourceoperazione. Per informazioni su come aggiungere un tag a un segreto, consulta the section called “Tag segreti ”.

UntagResource

Generato dall'UntagResourceoperazione. Per informazioni su come eliminare il tag di un segreto, consulta the section called “Tag segreti ”.

UpdateSecret

Generato dall'UpdateSecretoperazione. Per ulteriori informazioni sull'aggiornamento di un segreto, consulta the section called “Modificare un segreto”.

UpdateSecretVersionStage

Generato dall'UpdateSecretVersionStageoperazione. Per ulteriori informazioni sulle fasi di una versione, consulta the section called “Versioni segrete”.

ValidateResourcePolicy

Generato dall'ValidateResourcePolicyoperazione. Per informazioni sulle autorizzazioni, consulta the section called “Autenticazione e controllo degli accessi”.

Registra le voci del file di log da eliminare

Oltre agli eventi per le operazioni di Secrets Manager, Secrets Manager genera i seguenti eventi relativi alla cancellazione. Questi eventi hanno `"detail-type": ["AWS Service Event via CloudTrail"]`.

CancelSecretVersionDelete

Generato dal servizio Secrets Manager. Se chiami `DeleteSecret` su un segreto che ha delle versioni e successivamente chiami `RestoreSecret`, Secrets Manager registra nel file di log questo evento per ogni versione del segreto ripristinata. Per informazioni sul ripristino di un segreto eliminato, consulta the section called "Ripristino di un segreto".

EndSecretVersionDelete

Generato dal servizio Secrets Manager quando viene eliminata una versione del segreto. Per ulteriori informazioni, consulta the section called "Eliminare un segreto".

StartSecretVersionDelete

Generato dal servizio Secrets Manager quando Secrets Manager avvia l'eliminazione di una versione del segreto. Per informazioni sull'eliminazione dei segreti, consulta the section called "Eliminare un segreto".

SecretVersionDeletion

Generato dal servizio Secrets Manager quando Secrets Manager elimina una versione del segreto obsoleta. Per ulteriori informazioni, consulta Versioni del segreto.

Voci di log per la replica

Oltre agli eventi per le operazioni di Secrets Manager, Secrets Manager genera i seguenti eventi relativi alla replica. Questi eventi hanno `"detail-type": ["AWS Service Event via CloudTrail"]`.

ReplicationFailed

Generato dal servizio Secrets Manager quando la replica fallisce. Per ulteriori informazioni sulla replica di un segreto, consulta Replica in più regioni.

ReplicationStarted

Generato dal servizio Secrets Manager quando Secrets Manager inizia a replicare un segreto. Per ulteriori informazioni sulla replica di un segreto, consulta Replica in più regioni.

ReplicationSucceeded

Generato dal servizio Secrets Manager quando un segreto viene replicato correttamente. Per ulteriori informazioni sulla replica di un segreto, consulta Replica in più regioni.

Registra le voci del file di log per la rotazione

Oltre agli eventi per le operazioni di Secrets Manager, Secrets Manager genera i seguenti eventi relativi alla rotazione. Questi eventi hanno `"detail-type": ["AWS Service Event via CloudTrail"]`.

RotationStarted

Generato dal servizio Secrets Manager quando Secrets Manager inizia a ruotare un segreto. Per ulteriori informazioni sulla rotazione, consulta Rotazione dei segreti .

RotationAbandoned

Generato dal servizio Secrets Manager quando Secrets Manager abbandona un tentativo di rotazione e rimuove l'etichetta `AWSPENDING` da una versione esistente di un segreto. Secrets Manager abbandona la rotazione quando crei una nuova versione di un segreto durante la rotazione. Per ulteriori informazioni sulla rotazione, consulta Rotazione dei segreti .

RotationFailed

Generato dal servizio Secrets Manager quando la rotazione fallisce. Per ulteriori informazioni sulla rotazione, consulta the section called "Risoluzione dei problemi della rotazione".

RotationSucceeded

Generato dal servizio Secrets Manager quando un segreto viene ruotato correttamente. Per ulteriori informazioni sulla rotazione, consulta Rotazione dei segreti .

TestRotationStarted

Generato dal servizio Secrets Manager quando Secrets Manager inizia il test della rotazione per un segreto che non è pianificato per la rotazione immediata. Per ulteriori informazioni sulla rotazione, consulta Rotazione dei segreti .

TestRotationSucceeded

Generato dal servizio Secrets Manager quando Secrets Manager esegue correttamente il test della rotazione per un segreto che non è pianificato per la rotazione immediata. Per ulteriori informazioni sulla rotazione, consulta Rotazione dei segreti .

TestRotationFailed

Generato dal servizio Secrets Manager quando Secrets Manager esegue il test della rotazione per un segreto che non è pianificato per la rotazione immediata e la rotazione non riesce. Per ulteriori informazioni sulla rotazione, consulta the section called "Risoluzione dei problemi della rotazione".

Monitora Gestione dei segreti AWS con Amazon CloudWatch

Con Amazon CloudWatch, puoi monitorare AWS i servizi e creare allarmi per farti sapere quando le metriche cambiano. CloudWatch conserva queste statistiche per 15 mesi, in modo da poter accedere alle informazioni storiche e avere una prospettiva migliore sulle prestazioni della tua applicazione o del tuo servizio web. In Gestione dei segreti AWS effetti, puoi monitorare il numero di segreti nel tuo account, inclusi i segreti contrassegnati per l'eliminazione e le chiamate API a Secrets Manager, comprese le chiamate effettuate tramite la console. Per informazioni su come monitorare le metriche, consulta Utilizzare le CloudWatch metriche nella Guida per l'CloudWatch utente.

Per trovare le metriche di Secrets Manager

1. Sulla CloudWatch console, in Metriche, scegli Tutte le metriche.
2. Nella casella di ricerca Metriche, inserisci. `secret`
3. Esegui questa operazione:
   - Per monitorare il numero di segreti presenti nel tuo account, scegli AWS/SecretsManager, quindi seleziona SecretCount. Questa metrica viene pubblicata ogni ora.
   - Per monitorare le chiamate API a Secrets Manager, incluse le chiamate effettuate tramite la console, scegli Utilizzo > Per AWS risorsa, quindi seleziona le chiamate API da monitorare. Per un elenco di Secrets Manager APIs, consulta Operazioni di Secrets Manager.
4. Esegui questa operazione:
   - Per creare un grafico della metrica, consulta Graphing metrics nella Amazon CloudWatch User Guide.
   - Per rilevare anomalie, consulta Using CloudWatch anomaly detection nella Amazon CloudWatch User Guide.

- Per ottenere statistiche per una metrica, consulta Get statistics for a metric nella Amazon CloudWatch User Guide.

CloudWatch allarmi

Puoi creare un CloudWatch allarme che invia un messaggio Amazon SNS quando il valore di una metrica cambia e fa cambiare stato all'allarme. Puoi impostare un allarme sulla metrica Secrets Manager`ResourceCount`, che è il numero di segreti nel tuo account. Puoi anche impostare allarmi su Un allarme controlla una metrica in un periodo di tempo specificato ed esegue azioni in base al valore della metrica relativo a una determinata soglia in un certo numero di periodi di tempo. Gli allarmi richiamano azioni solo per cambiamenti di stato sostenuti. CloudWatch gli allarmi non richiamano azioni semplicemente perché si trovano in uno stato particolare; lo stato deve essere cambiato e mantenuto per un determinato numero di periodi.

Per ulteriori informazioni, consulta Usare gli CloudWatch allarmi Amazon e Creare un CloudWatch allarme basato sul rilevamento delle anomalie nella Guida per l'CloudWatch utente.

È anche possibile impostare allarmi che controllano determinate soglie e inviare notifiche o intraprendere azioni quando queste soglie vengono raggiunte. Per ulteriori informazioni, consulta la Amazon CloudWatch User Guide.

Abbina Gestione dei segreti AWS gli eventi con Amazon EventBridge

In Amazon EventBridge, puoi abbinare gli eventi di Secrets Manager dalle voci di CloudTrail registro. Puoi configurare EventBridge regole che cercano questi eventi e quindi inviano i nuovi eventi generati a un target affinché agisca. Per un elenco delle CloudTrail voci registrate da Secrets Manager, vedereCloudTrail voci. Per istruzioni sulla configurazione EventBridge, consulta Guida introduttiva EventBridge nella Guida per l'EventBridge utente.

Associa tutte le modifiche a un segreto specificato

> **Note**
>
> Poiché alcuni eventi di Secrets Manager restituiscono l'ARN del segreto con lettere maiuscole diverse, nei modelli di eventi che corrispondono a più di un'azione, per specificare un segreto

tramite ARN, potrebbe essere necessario includere sia le chiavi `arn` che `aRN`. Per ulteriori informazioni, consulta AWS re:Post.

L'esempio seguente mostra uno schema di EventBridge eventi che corrisponde alle voci di registro relative alle modifiche a un segreto.

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

Abbina gli eventi quando un valore segreto ruota

L'esempio seguente mostra uno schema di EventBridge eventi che corrisponde alle voci di CloudTrail registro per le modifiche ai valori segreti che si verificano in seguito agli aggiornamenti manuali o alla rotazione automatica. Poiché alcuni di questi eventi derivano dalle operazioni di Secrets Manager e altri sono generati dal servizio Secrets Manager, è necessario includere il `detail-type` per entrambi.

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

Monitora l'accesso ai Gestione dei segreti AWS segreti programmati per l'eliminazione

Puoi utilizzare una combinazione di AWS CloudTrail Amazon CloudWatch Logs e Amazon Simple Notification Service (Amazon SNS) per creare un allarme che ti avvisi di qualsiasi tentativo di accesso a un'eliminazione segreta in sospeso. Se ricevi una notifica mediante un allarme, puoi annullare l'eliminazione del segreto per avere più tempo per stabilire se vuoi effettivamente eliminarlo. La tua indagine potrebbe determinare il ripristino del segreto, se questo si rivela ancora effettivamente necessario. In alternativa, potrebbe essere necessario aggiornare l'utente con i dettagli del nuovo segreto da utilizzare.

Le seguenti procedure spiegano come ricevere una notifica quando una richiesta di `GetSecretValue` operazione genera un messaggio di errore specifico nei file di registro. CloudTrail Altre operazioni API possono essere eseguite sul segreto senza attivare l'allarme. Questo CloudWatch allarme rileva un utilizzo che potrebbe indicare che una persona o un'applicazione utilizza credenziali obsolete.

Prima di iniziare queste procedure, è necessario attivare l'account Regione AWS and CloudTrail in cui si intende monitorare Gestione dei segreti AWS le richieste API. Per istruzioni, consulta Creazione di un Trail per la prima volta nella AWS CloudTrail Guida per l'utente.

Passaggio 1: configurare la consegna dei file di CloudTrail registro a CloudWatch Logs

È necessario configurare la consegna dei file di CloudTrail registro ai CloudWatch registri. Lo fai in modo che CloudWatch Logs possa monitorarli per le richieste API di Secrets Manager per recuperare un segreto in attesa di eliminazione.

Per configurare la consegna dei file di CloudTrail registro a Logs CloudWatch

1. Apri la CloudTrail console all'indirizzo https://console.aws.amazon.com/cloudtrail/.
2. Nella barra di navigazione in alto, scegli Regione AWS per monitorare i segreti.
3. Nel riquadro di navigazione a sinistra, scegli Percorsi, quindi scegli il nome del percorso per cui configurare CloudWatch.
4. Nella pagina di configurazione dei percorsi, scorri verso il basso fino alla sezione CloudWatch Registri, quindi scegli l'icona di modifica (✏).

5. Per un New or existing log group (Gruppo nuovo o esistente di log), digita un nome per il gruppo di log, ad esempio **`CloudTrail/MyCloudWatchLogGroup`**.
6. Per il ruolo IAM, puoi utilizzare il ruolo predefinito denominato CloudTrail_ CloudWatchLogs _Role. Questo ruolo ha una politica di ruolo predefinita con le autorizzazioni necessarie per fornire CloudTrail eventi al gruppo di log.
7. Scegli Continue (Continua) per salvare la configurazione.
8. Nella pagina del gruppo di log CloudWatch Logs CloudTrail gli eventi associati all'attività delle API del tuo account AWS CloudTrail verranno inviati, scegli Consenti.

Fase 2: Creare l'allarme CloudWatch

Per ricevere una notifica quando un'operazione dell'`GetSecretValue`API Secrets Manager richiede di accedere a un segreto in attesa di eliminazione, è necessario creare un CloudWatch allarme e configurare la notifica.

Per creare un allarme CloudWatch

1. Accedi alla CloudWatch console all'indirizzo https://console.aws.amazon.com/cloudwatch/.
2. Nella barra di navigazione in alto, scegli la AWS regione in cui desideri monitorare i segreti.
3. Nel pannello di navigazione a sinistra, scegli Logs (Log).
4. Nell'elenco dei gruppi di log, seleziona la casella di controllo accanto al gruppo di log creato nella procedura precedente, ad esempio CloudTrail/MyCloudWatchLogGroup. Quindi scegli Create Metric Filter (Crea filtro parametro).
5. Per Filter Pattern (Modello filtri), digita o incolla quanto segue:

   ```
   { $.eventName = "GetSecretValue" && $.errorMessage = "*secret because it was marked
    for deletion*" }
   ```

   Scegli Assign Metric (Assegna parametro).
6. Nella pagina Create Metric Filter and Assign a Metric (Crea filtro parametri e assegna parametro), procedi nel seguente modo:

   a. Per Metric Namespace (Spazio dei nomi parametro), digita **`CloudTrailLogMetrics`**.

   b. Per Metric Name (Nome parametro) digita **`AttemptsToAccessDeletedSecrets`**.

c. Scegli Show advanced metric settings (Mostra impostazioni avanzate parametro) quindi, se necessario per Metric Value (Valore parametro) digita **1**.

d. Scegli Create Filter (Crea filtro).

7. Nella casella di filtro, scegli Create Alarm (Crea allarme).
8. Nella finestra Create Alarm (Crea allarme), procedi nel seguente modo:

a. In Name (Nome) digitare **`AttemptsToAccessDeletedSecretsAlarm`**.

b. In Whenever: (Ogni volta che:), per is: (è:), seleziona **>=**, quindi digita **1**.

c. Nel campo Send notification to: (Invia notifica a), procedi in uno dei seguenti modi:

- Per creare e utilizzare un nuovo argomento Amazon SNS, scegli New list (Nuovo elenco), quindi digita un nuovo nome argomento. Per Email list: (Elenco e-mail), digita almeno un indirizzo e-mail. È possibile digitare più di un indirizzo e-mail utilizzando la virgola come separatore.
- Per usare un argomento Amazon SNS esistente, scegli il nome dell'argomento da utilizzare. Se non esiste un elenco, seleziona Select list (Elenco di selezione).

d. Scegli Crea allarme.

Fase 3: Prova l' CloudWatch allarme

Per fare un test sull'allarme, crea un segreto e programmane l'eliminazione. Quindi, prova a recuperare il valore del segreto. Poco dopo riceverai un'e-mail all'indirizzo configurato nell'allarme. Questo avvisa di utilizzare un segreto pianificato per l'eliminazione.

Monitora Gestione dei segreti AWS i segreti per la conformità utilizzando AWS Config

Puoi usarle AWS Config per valutare i tuoi segreti per vedere se sono conformi ai tuoi standard. Puoi definire i requisiti interni di sicurezza e conformità per i segreti utilizzando AWS Config le regole. Quindi AWS Config puoi identificare i segreti che non sono conformi alle tue regole. Puoi anche tenere traccia delle modifiche ai metadati segreti, alla configurazione di rotazione, alla chiave KMS utilizzata per la crittografia segreta, alla funzione di rotazione Lambda e ai tag associati a un segreto.

Puoi configurare in modo che ti AWS Config avvisi delle modifiche. Per ulteriori informazioni, consulta l'argomento Notifiche AWS Config inviate a un Amazon SNS.

Se disponi di segreti in più Account AWS sedi e Regioni AWS nella tua organizzazione, puoi aggregare tali dati di configurazione e conformità. Per ulteriori informazioni, consulta Aggregazione di dati multiaccount e più regioni.

Per valutare se i segreti sono conformi

- Segui le istruzioni sulla valutazione delle tue risorse con AWS Config regole e scegli una delle seguenti regole:
  - `secretsmanager-secret-unused`— Controlla se sono stati eseguiti accessi ai segreti entro il numero specificato di giorni.
  - `secretsmanager-using-cmk`— Verifica se i segreti sono crittografati utilizzando la chiave Chiave gestita da AWS `aws/secretsmanager` o una chiave gestita dal cliente che hai creato. AWS KMS
  - `secretsmanager-rotation-enabled-check`— Verifica se la rotazione è configurata per i segreti memorizzati in Secrets Manager.
  - `secretsmanager-scheduled-rotation-success-check`: verifica se l'ultima rotazione riuscita rientra nella frequenza di rotazione configurata. La frequenza minima per la verifica è giornaliera.
  - `secretsmanager-secret-periodic-rotation`— Controlla se i segreti sono stati ruotati entro il numero specificato di giorni.

Monitora i costi di Secrets Manager

Puoi utilizzare Amazon CloudWatch per monitorare i Gestione dei segreti AWS costi stimati. Per ulteriori informazioni, consulta Creazione di un allarme di fatturazione per monitorare gli AWS addebiti stimati nella Guida per l'CloudWatch utente.

Un'altra opzione per monitorare i costi è il rilevamento delle anomalie AWS dei costi. Per ulteriori informazioni, consulta Rilevamento di spese insolite con AWS Cost Anomaly Detection nella AWS Cost Management User Guide.

Per informazioni sul monitoraggio dell'utilizzo di Secrets Manager, vedere the section called “Monitora con CloudWatch” ethe section called “Accedi con AWS CloudTrail ”.

Per informazioni sui Gestione dei segreti AWS prezzi, consultathe section called “Prezzi”.

Rileva le minacce con Amazon GuardDuty

Amazon GuardDuty è un servizio di rilevamento delle minacce che ti aiuta a proteggere account, container, carichi di lavoro e dati nel tuo AWS ambiente. Utilizzando modelli di machine learning (ML) e funzionalità di rilevamento di anomalie e minacce, monitora GuardDuty continuamente diverse fonti di log per identificare e dare priorità ai potenziali rischi per la sicurezza e alle attività dannose nel tuo ambiente. Ad esempio, GuardDuty rileverà potenziali minacce come l'accesso insolito o sospetto ai segreti e l'esfiltrazione di credenziali nel caso in cui rilevi credenziali create esclusivamente per un' EC2 istanza Amazon tramite un ruolo di avvio dell'istanza ma utilizzate da un altro account all'interno. AWS Per ulteriori informazioni, consulta la Amazon GuardDuty User Guide.

Un altro esempio di utilizzo del rilevamento è il comportamento anomalo. Ad esempio, se Gestione dei segreti AWS in genere riceve `create-secret` `get-secret-value``describe-secret`, e `list-secrets` chiamate da un'entità che utilizza Java SDK e quindi un'entità diversa inizia a chiamare `batch-get-secret-value` e `get-secret-value` utilizzare l'oggetto AWS CLI dall'esterno della VPN, è GuardDuty possibile segnalare un risultato che la seconda entità sta invocando in modo anomalo. APIs Per ulteriori informazioni, consulta GuardDuty IAM finding type:/. CredentialAccess IAMUser AnomalousBehavior

Convalida della conformità per Gestione dei segreti AWS

La vostra responsabilità di conformità quando utilizzate Secrets Manager è determinata dalla sensibilità dei vostri dati, dagli obiettivi di conformità dell'azienda e dalle leggi e dai regolamenti applicabili. AWS fornisce le seguenti risorse per contribuire alla conformità:

- Le guide rapide per la sicurezza e la conformità: queste guide all'implementazione illustrano considerazioni architettoniche e forniscono i passaggi per l'implementazione di ambienti di base incentrati sulla sicurezza e sulla conformità su AWS.
- Whitepaper sull'architettura per la sicurezza e la conformità HIPAA: questo white paper descrive come le aziende possono utilizzare per creare applicazioni conformi allo standard HIPAA. AWS
- AWS Risorse per la conformità Risorse per la conformità: questa raccolta di potrebbe riguardare il settore e la località in cui operate.
- AWS Config valuta il livello di conformità delle configurazioni delle risorse con le pratiche interne, le linee guida e i regolamenti di settore. Per ulteriori informazioni, consulta the section called "Monitora i segreti ai fini della conformità".
- AWS Security Hub CSPMfornisce una visione completa dello stato di sicurezza dell'utente AWS che consente di verificare la conformità agli standard e alle best practice del settore della sicurezza. Per informazioni sull'utilizzo di Security Hub CSPM per valutare le risorse di Secrets Manager, consulta Gestione dei segreti AWS i controlli nella Guida per l'AWS Security Hub CSPM utente.
- IAM Access Analyzer analizza le policy, incluse le istruzioni di condizione in una policy, che consentono a un'entità esterna di accedere a un segreto. Per ulteriori informazioni, consulta Anteprima dell'accesso con Access Analyzer.
- AWS Systems Manager fornisce runbook predefiniti per Secrets Manager. Per ulteriori informazioni, consulta Systems Manager Automation runbook reference for Secrets Manager (Riferimento del runbook di automazione di Systems Manager).
- È possibile scaricare i report di controllo di terze parti utilizzando. AWS Artifact Per ulteriori informazioni, consulta Scaricamento dei report in AWS Artifact .

Standard di conformità

Gestione dei segreti AWS è stato sottoposto a revisione per i seguenti standard e può far parte della vostra soluzione quando è necessario ottenere la certificazione di conformità.

- HIPAA — AWS ha ampliato il suo programma di conformità all'Health Insurance Portability and Accountability Act (HIPAA) per includerlo Gestione dei segreti AWS come servizio idoneo all'HIPAA. Se hai sottoscritto un Business Associate Agreement (BAA) con AWS, puoi usare Secrets Manager per aiutarti a creare le tue applicazioni conformi allo standard HIPAA. AWS offre un white paper incentrato sull'HIPAA per i clienti interessati a saperne di più su come sfruttare per l'elaborazione e l'archiviazione delle informazioni sanitarie. AWS Per ulteriori informazioni, consulta Compliance HIPAA.
- Organizzazione partecipante allo standard PCI: Gestione dei segreti AWS dispone di un attestato di conformità allo standard di sicurezza dei dati (DSS) PCI (Payment Card Industry) versione 3.2 al livello 1 del service provider. I clienti che utilizzano AWS prodotti e servizi per archiviare, elaborare o trasmettere i dati dei titolari di carte possono utilizzarli Gestione dei segreti AWS mentre gestiscono la propria certificazione di conformità PCI DSS. Per ulteriori informazioni su PCI DSS, incluso come richiedere una copia del PCI AWS Compliance Package, vedere PCI DSS Level 1.
- ISO: Gestione dei segreti AWS ha completato con successo la certificazione di conformità per ISO/IEC 27001, ISO/IEC 27017, 27018 e ISO 9001. ISO/IEC Per ulteriori informazioni, consulta ISO 27001, ISO 27017, ISO 27018, ISO 9001.
- I report AICPA SOC — System and Organization Control (SOC) sono rapporti di esame indipendenti di terze parti che dimostrano come Secrets Manager raggiunge i controlli e gli obiettivi chiave di conformità. Lo scopo di questi report è aiutare voi e i vostri revisori a comprendere i AWS controlli che vengono stabiliti per supportare le operazioni e la conformità. Per ulteriori informazioni, consulta la pagina Conformità SOC.
- FedRAMP — Il Federal Risk and Authorization Management Program (FedRAMP) è un programma a livello governativo che fornisce un approccio standardizzato alla valutazione della sicurezza, all'autorizzazione e al monitoraggio continuo per prodotti e servizi cloud. Il programma FedRAMP fornisce anche autorizzazioni provvisorie per servizi e regioni East/West per e per l'utilizzo di dati governativi o regolamentati. GovCloud Per ulteriori informazioni, consulta Conformità FedRAMP.
- Dipartimento della Difesa — La Guida ai requisiti di sicurezza del cloud computing (SRG) del Dipartimento della Difesa (DoD) fornisce un processo di valutazione e autorizzazione standardizzato per i fornitori di servizi cloud (CSPs) per ottenere un'autorizzazione provvisoria del DoD, in modo che possano servire i clienti del DoD. Per ulteriori informazioni, consulta DoD SRG Resources
- IRAP — L'Information Security Registered Assessors Program (IRAP) consente ai clienti del governo australiano di verificare l'esistenza di controlli appropriati e determinare il modello di responsabilità appropriato per soddisfare i requisiti dell'Information Security Manual (ISM)

del governo australiano prodotto dall'Australian Cyber Security Centre (ACSC). Per ulteriori informazioni, consulta IRAP Resources

- OSPAR — Amazon Web Services (AWS) ha ottenuto l'attestazione OSPAR (Outsourced Service Provider's Audit Report). AWS l'allineamento con le linee guida dell'Association of Banks in Singapore (ABS) sugli obiettivi e le procedure di controllo per i fornitori di servizi in outsourcing (ABS Guidelines) dimostra ai clienti AWS l'impegno a soddisfare le elevate aspettative per i fornitori di servizi cloud stabilite dal settore dei servizi finanziari a Singapore. Per ulteriori informazioni, consulta Risorse OSPAR.

Sicurezza in Gestione dei segreti AWS

La sicurezza AWS è la massima priorità. In qualità di AWS cliente, puoi beneficiare di un data center e di un'architettura di rete progettati per soddisfare i requisiti delle organizzazioni più sensibili alla sicurezza.

Tu e io AWS condividiamo la responsabilità della sicurezza. Il modello di responsabilità condivisa descrive questo come sicurezza del cloud e sicurezza nel cloud:

- Sicurezza del cloud: AWS è responsabile della protezione dell'infrastruttura che gestisce AWS i servizi nel AWS cloud. AWS ti offre anche servizi che puoi utilizzare in modo sicuro. I revisori di terze parti testano e verificano regolarmente l'efficacia della sicurezza come parte dei programmi di conformitàAWS. Per ulteriori informazioni sui programmi di conformità applicabili Gestione dei segreti AWS, consulta AWS Services in Scope by Compliance Program.
- Sicurezza nel cloud: il tuo AWS servizio determina la tua responsabilità. L'utente è anche responsabile di altri fattori, tra cui la riservatezza dei dati, i requisiti della propria azienda e le leggi e normative vigenti.

Per ulteriori risorse, consulta Security Pillar — AWS Well-Architected Framework.

Argomenti

Riduci i rischi derivanti dall'utilizzo di per archiviare i tuoi AWS CLI segreti Gestione dei segreti AWS

Quando usate il AWS Command Line Interface (AWS CLI) per richiamare AWS le operazioni, inserite tali comandi in una shell di comando. Ad esempio, è possibile utilizzare il prompt dei comandi di Windows o Windows PowerShell oppure la shell Bash o Z, tra le altre. Molte di queste shell di comando includono funzionalità progettate per aumentare la produttività. Ma queste funzionalità possono essere utilizzate per compromettere i tuoi segreti. Ad esempio, nella maggior parte delle shell puoi utilizzare il tasto freccia verso l'alto per visualizzare l'ultimo comando inserito. La funzionalità di cronologia dei comandi può essere sfruttata da chiunque acceda alla tua sessione non protetta. Inoltre, altre utility che funzionano in background potrebbero accedere ai parametri di comando, con l'obiettivo di aiutarti a eseguire le tue attività in modo più efficace. Per ridurre tali rischi, accertati di procedere nel modo seguente:

- Blocca sempre il tuo computer quando ti allontani dalla console.
- Disinstallare o disattivare le utility della console non necessarie o non più utilizzate.
- Assicurarsi che la shell o il programma di accesso remoto, se viene utilizzato l'uno o l'altro, non registrino i comandi digitati.
- Utilizzare le tecniche per inviare parametri non acquisiti dalla cronologia dei comandi della shell. L'esempio seguente mostra come digitare il testo segreto in un file di testo, quindi passare il file al Gestione dei segreti AWS comando e distruggerlo immediatamente. Ciò significa che la tipica cronologia dei comandi della shell non cattura il testo del segreto.

  L'esempio seguente mostra i comandi tipici di Linux, ma la tua shell potrebbe prevedere comandi lievemente diversi:

```
$ 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.
```

Dopo l'esecuzione di questi comandi, dovresti essere in grado di usare le frecce verso l'alto e il basso per scorrere la cronologia dei comandi e vedere che il testo del segreto non è presente in alcuna riga.

> **Important**
>
> Per impostazione predefinita, non puoi adottare una tecnica equivalente in Windows a meno che prima tu non riduca a 1 le dimensioni del buffer della cronologia dei comandi.

Per configurare il prompt dei comandi di Windows affinché presenti solo 1 buffer della cronologia dei comandi di 1 comando

1. Aprire un prompt dei comandi amministratore (Run as administrator (Esegui come amministratore)).
2. Scegliete l'icona in alto a sinistra, quindi scegliete Proprietà.
3. Nella scheda Options (Opzioni) imposta Buffer Size (Dimensioni buffer) e Number of Buffers (Numero di buffer) entrambi su **1**, quindi scegli OK.
4. Ogni volta che devi digitare un comando che non vuoi includere nella cronologia, fallo seguire immediatamente da un altro comando, ad esempio:

   ```
   echo.
   ```

   In questo modo sei sicuro che il comando sensibile non venga incluso.

Per la shell del prompt dei comandi di Windows, puoi scaricare lo SysInternals SDeletestrumento e quindi utilizzare comandi simili ai seguenti:

```
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.
```

Autenticazione e controllo degli accessi per Gestione dei segreti AWS

Utilizza Secrets ManagerAWS Identity and Access Management (IAM) per proteggere l'accesso ai segreti. IAM fornisce autenticazione e controllo degli accessi. Autenticazione verifica l'identità di coloro che effettuano le richieste. Secrets Manager utilizza un processo di accesso con le password, le chiavi di accesso e l'autenticazione a più fattori (MFA) per verificare l'identità degli utenti. Vedi Accesso a AWS. Autorizzazione assicura che solo gli individui approvati possano eseguire operazioni sulle risorse AWS ad esempio sui segreti. Secrets Manager utilizza le policy per definire chi ha accesso a quali risorse e quali azioni l'identità può intraprendere su tali risorse. Vedere Autorizzazioni e policy in IAM.

Argomenti

- Riferimento alle autorizzazioni per Gestione dei segreti AWS
- Autorizzazioni di amministrazione di Secrets Manager
- Autorizzazioni per accedere ai segreti
- Autorizzazioni per le funzioni di rotazione Lambda
- Autorizzazioni per le chiavi di crittografia
- Autorizzazioni per la replica
- Policy basate sull'identità
- Policy basate sulle risorse
- Controlla l'accesso ai segreti utilizzando il controllo degli accessi basato sugli attributi (ABAC)
- AWS politica gestita per Gestione dei segreti AWS
- Determina chi dispone delle autorizzazioni per accedere ai tuoi segreti Gestione dei segreti AWS
- Accedi ai Gestione dei segreti AWS segreti da un altro account

- Accedi ai segreti da un ambiente locale

Riferimento alle autorizzazioni per Gestione dei segreti AWS

Il riferimento alle autorizzazioni per Secrets Manager è disponibile nella sezione Azioni, risorse e chiavi di condizione del Service Authorization Reference. Gestione dei segreti AWS

Autorizzazioni di amministrazione di Secrets Manager

Per concedere le autorizzazioni di amministratore di Secrets Manager, seguire le istruzioni riportate in Aggiunta e rimozione di autorizzazioni per identità IAM, e allegare i seguenti criteri:

- SecretsManagerReadWrite
- IAMFullAccess

Si consiglia di non concedere autorizzazioni di amministratore agli utenti finali. Sebbene ciò consente agli utenti di creare e gestire i propri segreti, l'autorizzazione necessaria per abilitare la rotazione (IAMFullAccess) concede autorizzazioni significative che non sono appropriate per gli utenti finali.

Autorizzazioni per accedere ai segreti

Utilizzando le policy di autorizzazione IAM, puoi controllare quali utenti o servizi possono accedere ai segreti. Una policy di autorizzazioni descrive chi può eseguire quali azioni su quali risorse. Puoi:

- the section called "Policy basate sull'identità"
- the section called "Policy basate sulle risorse"

Autorizzazioni per le funzioni di rotazione Lambda

Secrets Manager utilizza AWS Lambda funzioni per ruotare i segreti. La funzione Lambda deve avere accesso al segreto e al database o al servizio per cui il segreto contiene le credenziali. Per informazioni, consulta Autorizzazioni per la rotazione.

Autorizzazioni per le chiavi di crittografia

Secrets Manager utilizza le chiavi AWS Key Management Service (AWS KMS) per crittografare i segreti. Dispone Chiave gestita da AWS `aws/secretsmanager` automaticamente delle autorizzazioni corrette. Se si utilizza una chiave KMS diversa, Secrets Manager necessita delle

autorizzazioni per tale chiave. Per informazioni, consulta the section called "Autorizzazioni per la chiave KMS".

Autorizzazioni per la replica

Utilizzando le policy di autorizzazione IAM, puoi controllare quali utenti o servizi possono replicare i tuoi segreti in altre regioni. Per informazioni, consulta the section called "Impedire la replica".

Policy basate sull'identità

Come allegare policy di autorizzazioni a identità, utenti, gruppi, ruoli, servizi e risorse IAM. In una policy basata sull'identità, specifichi a quali segreti può accedere l'identità e le azioni che l'identità può eseguire sui segreti. Per ulteriori informazioni, consulta Aggiunta e rimozione di autorizzazioni per identità IAM.

Puoi concedere autorizzazioni a un ruolo che rappresenta un'applicazione o un utente in un altro servizio. Ad esempio, un'applicazione in esecuzione su un'istanza Amazon EC2 potrebbe dover accedere a un database. È possibile creare un ruolo IAM associato al profilo dell'istanza EC2 e quindi utilizzare una policy di autorizzazioni per concedere al ruolo l'accesso al segreto che contiene le credenziali per il database. Per ulteriori informazioni, consulta Utilizzo di un ruolo IAM per concedere autorizzazioni ad applicazioni in esecuzione su istanze di Amazon EC2. Altri servizi a cui è possibile allegare i ruoli includono Amazon Redshift, AWS Lambda ed Amazon ECS.

Puoi concedere autorizzazioni agli utenti autenticati da un sistema di identità diverso da IAM. Ad esempio, è possibile associare ruoli IAM; a utenti che utilizzano app per dispositivi mobili e che effettuano l'accesso con Amazon Cognito. Il ruolo concede all'app le credenziali provvisorie con le autorizzazioni nella policy di autorizzazioni del ruolo. È quindi possibile utilizzare una policy di autorizzazione per concedere al ruolo l'accesso al segreto. Per ulteriori informazioni, consulta Provider di identità e federazione.

Si possono utilizzare policy basate su identità per:

- Concedi un accesso alle identità a più segreti.
- Controllare chi può creare nuovi segreti e chi può accedere a segreti che non sono ancora stati creati.
- Concedi un accesso a un gruppo IAM ai segreti.

Esempi:

- Esempio: Autorizzazione per recuperare valori segreti individuali
- Esempio: autorizzazione a leggere e descrivere singoli segreti
- Esempio: autorizzazione a recuperare un gruppo di valori segreti in un batch
- Esempio: il carattere jolly
- Esempio: Autorizzazione per creare segreti
- Esempio: nega una AWS KMS chiave specifica per crittografare i segreti

Esempio: Autorizzazione per recuperare valori segreti individuali

Per concedere il permesso di recuperare valori segreti, è possibile allegare policy a segreti o identità. Per informazioni sul tipo di criterio da utilizzare, vedere Policy basate su identità e policy basate su risorse. Per informazioni sul collegamento di una policy a un'identità, vedere the section called "Policy basate sulle risorse" e the section called "Policy basate sull'identità".

Questo esempio è utile quando si desidera concedere l'accesso a un gruppo IAM. Per concedere l'autorizzazione a recuperare un gruppo di segreti in una chiamata API batch, consulta the section called "Esempio: autorizzazione a recuperare un gruppo di valori segreti in un batch".

Example Leggi un segreto crittografato utilizzando una chiave gestita dal cliente

Se un segreto è crittografato utilizzando una chiave gestita dal cliente, puoi concedere l'accesso alla lettura del segreto allegando la seguente politica a un'identità. \

JSON

```
{
"Version":"2012-10-17",
"Statement": [
{
  "Effect": "Allow",
  "Action": "secretsmanager:GetSecretValue",
  "Resource": "arn:aws:secretsmanager:us-east-1:123456789012:secret:secretName-
AbCdEf"
},
{
  "Effect": "Allow",
  "Action": "kms:Decrypt",
  "Resource": "arn:aws:kms:us-east-1:123456789012:key/key-id"
}
```

```
]
}
```

Esempio: autorizzazione a leggere e descrivere singoli segreti

Example Leggi e descrivi un segreto

È possibile concedere l'accesso a un segreto allegando a un'identità la policy seguente.

JSON

```
{
"Version":"2012-10-17",
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "secretsmanager:GetSecretValue",
      "secretsmanager:DescribeSecret"
      ],
      "Resource": "arn:aws:secretsmanager:us-
east-1:123456789012:secret:secretName-AbCdEf"
    }
]
}
```

Esempio: autorizzazione a recuperare un gruppo di valori segreti in un batch

Example Leggi un gruppo di segreti in un batch

Allegando a un'identità la policy seguente, è possibile concedere l'accesso per recuperare un gruppo di segreti in una chiamata API batch. La policy limita il chiamante in modo che possa recuperare solo i segreti specificati da*SecretARN1*, e *SecretARN2SecretARN3*, anche se la chiamata batch include altri segreti. Se il chiamante richiede anche altri segreti nella chiamata API batch, Secrets Manager non li restituirà. Per ulteriori informazioni, vedere. `BatchGetSecretValue` .

JSON

```
{
```

```
"Version":"2012-10-17",
"Statement": [
{
  "Effect": "Allow",
  "Action": [
    "secretsmanager:BatchGetSecretValue",
    "secretsmanager:ListSecrets"
  ],
  "Resource": "*"
},
{
  "Effect": "Allow",
  "Action": [
    "secretsmanager:GetSecretValue"
  ],
  "Resource": [
    "arn:aws:secretsmanager:us-east-1:123456789012:secret:secretName1-AbCdEf",
    "arn:aws:secretsmanager:us-east-1:123456789012:secret:secretName2-AbCdEf",
    "arn:aws:secretsmanager:us-east-1:123456789012:secret:secretName3-AbCdEf"
    ]
}
]
}
```

Esempio: il carattere jolly

È possibile utilizzare caratteri jolly per includere un set di valori in un elemento della policy.

Example Accedi a tutti i segreti di un percorso

La seguente politica consente l'accesso per recuperare tutti i segreti il cui nome inizia con "»*TestEnv/*.

JSON

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

```
    }
}
```

Example Accedi ai metadati relativi a tutti i segreti

Le seguenti sovvenzioni `DescribeSecret` e le autorizzazioni che iniziano con `List`: `ListSecrets` e `ListSecretVersionIds`.

JSON

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

Example Corrisponde al nome segreto

La policy seguente concede tutte le autorizzazioni di Secrets Manager per un segreto, per nome. Per utilizzare questa policy, consultare the section called "Policy basate sull'identità".

Per abbinare un nome segreto, creare l'ARN per il segreto mettendo insieme la regione, l'ID dell'Account, il nome segreto e il carattere jolly (?) per abbinare singoli caratteri casuali. Secrets Manager aggiunge sei caratteri casuali ai nomi segreti come parte del loro ARN, per poter utilizzare questo carattere jolly per abbinare tali caratteri. Se si utilizza la sintassi `"another_secret_name-*"`, Secret Manager individua la corrispondenza non solo del determinato segreto con i 6 caratteri casuali, ma anche di `"another_secret_name-<anything-here>a1b2c3"`.

Perché puoi prevedere tutte le parti dell'ARN di un segreto eccetto i 6 caratteri casuali, usando il carattere jolly `'??????'` la sintassi ti consente di concedere le autorizzazioni in modo sicuro a un segreto che non esiste ancora. Tuttavia, tieni presente che, se elimini il segreto e lo ricrei con lo stesso nome, l'utente riceve automaticamente l'autorizzazione per il nuovo segreto, anche se i 6 caratteri sono cambiati.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "secretsmanager:*",
            "Resource": [
                "arn:aws:secretsmanager:us-
east-1:123456789012:secret:a_specific_secret_name-a1b2c3",
                "arn:aws:secretsmanager:us-
east-1:123456789012:secret:another_secret_name-??????"
            ]
        }
    ]
}
```

Esempio: Autorizzazione per creare segreti

Per concedere a un utente le autorizzazioni per creare un segreto, allegare una policy di autorizzazioni a un gruppo IAM a cui appartiene l'utente. Consultare Gruppi di utenti IAM.

Example Crea segreti

La policy seguente concede l'autorizzazione per creare segreti e visualizzare un elenco di segreti. Per utilizzare questa policy, consultare the section called "Policy basate sull'identità".

JSON

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

```
}
]
}
```

Esempio: nega una AWS KMS chiave specifica per crittografare i segreti

> **Important**
>
> Per negare una chiave gestita dal cliente, ti consigliamo di limitare l'accesso utilizzando una politica o una concessione di chiavi. Per ulteriori informazioni, consulta Autenticazione e controllo degli accessi AWS KMS nella Guida per gli sviluppatori di AWS Key Management Service .

Example Nega la chiave gestita AWS **`aws/secretsmanager`**

La seguente politica nega l'uso di Chiave gestita da AWS `aws/secretsmanager` per la creazione o l'aggiornamento di segreti. Questa politica richiede che i segreti vengano crittografati utilizzando una chiave gestita dal cliente. La policy include due dichiarazioni:

1. La prima dichiarazione`Sid: "RequireCustomerManagedKeysOnSecrets"`, nega le richieste di creazione o aggiornamento di segreti utilizzando. Chiave gestita da AWS `aws/secretsmanager`
2. La seconda istruzione`Sid: "RequireKmsKeyIdParameterOnCreate"`, nega le richieste di creazione di segreti che non includono una chiave KMS, poiché Secrets Manager utilizzerebbe per impostazione predefinita il. Chiave gestita da AWS `aws/secretsmanager`

JSON

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

```
            "secretsmanager:UpdateSecret"
        ],
        "Resource": "*",
        "Condition": {
            "StringLikeIfExists": {
                "secretsmanager:KmsKeyArn": "<key_ARN_of_the_AWS_managed_key>"
            }
        }
    },
    {
        "Sid": "RequireKmsKeyIdParameterOnCreate",
        "Effect": "Deny",
        "Action": "secretsmanager:CreateSecret",
        "Resource": "*",
        "Condition": {
            "Null": {
                "secretsmanager:KmsKeyArn": "true"
            }
        }
    }
]
}
```

Policy basate sulle risorse

In una policy basata sulle risorse, si specifica chi può accedere al segreto e le azioni che è possibile eseguire sul segreto. Le policy basate su risorse possono essere utilizzate per:

- Concedre l'accesso a più utenti o ruoli ad un singolo segreto.
- Concedi l'accesso a utenti o ruoli in altri AWS account.

Quando si allega una policy basata sulle risorse a un segreto nella console, Secrets Manager utilizza il motore di ragionamento automatico Zelkova e l'API `ValidateResourcePolicy` per impedirti di concedere a una vasta gamma di entità IAM l'accesso ai tuoi segreti. In alternativa, è possibile chiamare `PutResourcePolicy` API con `BlockPublicPolicy` parametro da CLI o SDK.

> **Important**
>
> La convalida della politica delle risorse e del `BlockPublicPolicy` parametro aiutano a proteggere le risorse impedendo che l'accesso pubblico venga concesso attraverso le

politiche relative alle risorse direttamente collegate ai segreti dell'utente. Oltre a utilizzare queste funzionalità, esamina attentamente le seguenti politiche per verificare che non garantiscano l'accesso pubblico:

- Politiche basate sull'identità collegate ai AWS principali associati (ad esempio, ruoli IAM)
- Politiche basate sulle risorse collegate alle AWS risorse associate (ad esempio, () chiavi) AWS Key Management Service AWS KMS

Per rivedere le autorizzazioni relative ai tuoi segreti, consulta. Determinazione di chi ha le autorizzazioni per i segreti

Come visualizzare, modificare o eliminare la policy di risorse per un segreto (console)

1. Apri la console Secrets Manager all'indirizzo https://console.aws.amazon.com/secretsmanager/.
2. Dall'elenco dei segreti, scegli il segreto.
3. Nella pagina dei dettagli del segreto, nella sezione Panoramica, nella sezione Autorizzazioni risorse, scegli Modifica autorizzazioni.
4. Nel campo Codice, eseguire una delle operazioni seguenti, quindi scegliere Save (Salva):
   - Per allegare o modificare una policy delle risorse, immettere la policy.
   - Per eliminare la policy, deselezionare il campo del codice.

AWS CLI

Example Recupero di una politica sulle risorse

L'esempio di `get-resource-policy` seguente mostra come recuperare la policy basata su risorse collegata a un segreto.

```
aws secretsmanager get-resource-policy \
    --secret-id MyTestSecret
```

Example Eliminazione di una policy sulle risorse

L'esempio di `delete-resource-policy` seguente mostra come eliminare la policy basata su risorse collegata a un segreto.

```
aws secretsmanager delete-resource-policy \
    --secret-id MyTestSecret
```

Example Aggiunta di una policy sulle risorse

L'esempio di `put-resource-policy` seguente mostra come aggiungere una policy di autorizzazioni a un segreto, verificando innanzitutto che la policy non fornisca un accesso ampio al segreto. La policy viene letta da un file. Per ulteriori informazioni, consulta Caricamento AWS CLI dei parametri da un file nella Guida AWS CLI per l'utente.

```
aws secretsmanager put-resource-policy \
    --secret-id MyTestSecret \
    --resource-policy file://mypolicy.json \
    --block-public-policy
```

Contenuto di `mypolicy.json`:

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::123456789012:role/MyRole"
            },
            "Action": "secretsmanager:GetSecretValue",
            "Resource": "*"
        }
    ]
}
```

AWS SDK

Per recuperare la policy collegata a un segreto, utilizzare `GetResourcePolicy`.

Per eliminare una policy collegata a un segreto, utilizzare `DeleteResourcePolicy`.

Per collegare una policy a un segreto, utilizzare `PutResourcePolicy`. Se è già associata una policy, il comando la sostituisce con la nuova policy. La policy deve essere formattata come testo strutturato JSON. Vedere Struttura dei documenti di policy JSON.

Per ulteriori informazioni, consulta the section called "AWS SDKs".

Esempi

Esempi:

- Esempio: Autorizzazione per recuperare valori segreti individuali
- Esempio: autorizzazioni e VPCs
- Esempio: Principale del servizio

Esempio: Autorizzazione per recuperare valori segreti individuali

Per concedere il permesso di recuperare valori segreti, è possibile allegare policy a segreti o identità. Per informazioni sul tipo di criterio da utilizzare, vedere Policy basate su identità e policy basate su risorse. Per informazioni sul collegamento di una policy a un'identità, vedere the section called "Policy basate sulle risorse" e the section called "Policy basate sull'identità".

Questo esempio è utile quando si desidera concedere l'accesso a un singolo segreto a più utenti o ruoli. Per concedere l'autorizzazione a recuperare un gruppo di segreti in una chiamata API batch, consulta the section called "Esempio: autorizzazione a recuperare un gruppo di valori segreti in un batch".

Example Leggi un segreto

È possibile concedere l'accesso a un segreto allegando a tale segreto la policy seguente.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:role/EC2RoleToAccessSecrets"
```

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

Esempio: autorizzazioni e VPCs

Se è necessario accedere a Secrets Manager da un VPC, è possibile assicurarsi che le richieste a Secrets Manager provengano dal VPC includendo una condizione nelle policy di autorizzazione. Per ulteriori informazioni, consultare Limita le richieste con le condizioni degli endpoint VPC e the section called "Endpoint VPC (AWS PrivateLink)".

Assicurati che le richieste di accesso al segreto provenienti da altri AWS servizi provengano anche dal VPC, altrimenti questa politica negherà loro l'accesso.

Example Richiedi che le richieste arrivino tramite un endpoint VPC

La seguente policy consente all'utente di eseguire operazioni Secret Manager solo quando la richiesta proviene tramite l'endpoint VPC specificato *vpce-1234a5678b9012c*.

JSON

```
{
"Id": "example-policy-1",
"Version":"2012-10-17",
"Statement": [
{
  "Sid": "RestrictGetSecretValueoperation",
  "Effect": "Deny",
  "Principal": "*",
  "Action": "secretsmanager:GetSecretValue",
  "Resource": "*",
  "Condition": {
    "StringNotEquals": {
      "aws:sourceVpce": "vpce-12345678"
    }
  }
}
]
```

```
}
```

Example Richiedi che le richieste provengano da un VPC

I seguenti comandi della policy consentono di creare e gestire i segreti solo quando la loro provenienza è *vpc-12345678*. Inoltre, la policy consente le operazioni che utilizzano il valore del segreto crittografato solo quando le richieste provengono da vpc-2b2b2b2b. Puoi usare una policy come questa se esegui un'applicazione in un VPC, ma utilizzi un secondo VPC separato per le funzioni di gestione.

JSON

```
{
"Id": "example-policy-2",
"Version":"2012-10-17",
"Statement": [
{
  "Sid": "AllowAdministrativeActionsfromONLYvpc-12345678",
  "Effect": "Deny",
  "Principal": "*",
  "Action": [
    "secretsmanager:Create*",
    "secretsmanager:Put*",
    "secretsmanager:Update*",
    "secretsmanager:Delete*",
    "secretsmanager:Restore*",
    "secretsmanager:RotateSecret",
    "secretsmanager:CancelRotate*",
    "secretsmanager:TagResource",
    "secretsmanager:UntagResource"
  ],
  "Resource": "*",
  "Condition": {
    "StringNotEquals": {
      "aws:sourceVpc": "vpc-12345678"
    }
  }
},
{
  "Sid": "AllowSecretValueAccessfromONLYvpc-2b2b2b2b",
  "Effect": "Deny",
```

```
  "Principal": "*",
  "Action": [
    "secretsmanager:GetSecretValue"
  ],
  "Resource": "*",
  "Condition": {
    "StringNotEquals": {
      "aws:sourceVpc": "vpc-2b2b2b2b"
    }
  }
}
]
}
```

Esempio: Principale del servizio

Se la policy relativa alle risorse allegata al tuo segreto include un AWS service principal, ti consigliamo di utilizzare le chiavi di condizione SourceAccount globali aws: SourceArn e aws:. I valori ARN e account sono inclusi nel contesto di autorizzazione solo quando una richiesta arriva a Secrets Manager da un altro servizio AWS . Questa combinazione di condizioni evita un potenziale confused deputy scenario (scenario "deputy confused").

Se una risorsa ARN include caratteri non consentiti in una policy di risorse, non potrai utilizzare l'ARN di tale risorsa nel valore della chiave di condizione `aws:SourceArn`. Devi invece utilizzare la chiave di condizione `aws:SourceAccount`. Per ulteriori informazioni, consulta Requisiti IAM.

I service principal non vengono in genere utilizzati come responsabili in una policy allegata a un segreto, ma alcuni AWS servizi lo richiedono. Per informazioni sulle policy delle risorse che un servizio richiede di allegare a un segreto, consultare la documentazione del servizio.

Example Consenti a un servizio di accedere a un segreto utilizzando un service principal

JSON

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

```
    "Service": [
      "s3.amazonaws.com"
    ]
  },
  "Action": "secretsmanager:GetSecretValue",
  "Resource": "*",
  "Condition": {
    "ArnLike": {
      "aws:sourceArn": "arn:aws:s3::123456789012:*"
    },
    "StringEquals": {
      "aws:sourceAccount": "123456789012"
    }
  }

}
]
}
```

Controlla l'accesso ai segreti utilizzando il controllo degli accessi basato sugli attributi (ABAC)

Il controllo degli accessi basato sugli attributi (ABAC) è una strategia di autorizzazione che definisce le autorizzazioni in base agli attributi o alle caratteristiche dell'utente, dei dati o dell'ambiente, ad esempio il reparto, l'unità aziendale o altri fattori che potrebbero influire sull'esito dell'autorizzazione. In AWS, questi attributi sono chiamati tag.

Usare i tag per controllare le autorizzazioni è utile in ambienti soggetti a una rapida crescita e aiuta in situazioni in cui la gestione delle policy diventa complicata. Le regole ABAC vengono valutate dinamicamente in fase di esecuzione, il che significa che l'accesso degli utenti alle applicazioni e ai dati e il tipo di operazioni consentite cambiano automaticamente in base ai fattori contestuali della politica. Ad esempio, se un utente cambia reparto, l'accesso viene regolato automaticamente senza la necessità di aggiornare le autorizzazioni o richiedere nuovi ruoli. Per ulteriori informazioni, consulta: A cosa serve ABAC? AWS, Definisci le autorizzazioni per accedere ai segreti in base ai tag. e scalate le vostre esigenze di autorizzazione per Secrets Manager utilizzando ABAC con IAM Identity Center.

Esempio: consenti a un'identità di accedere ai segreti con tag specifici

La seguente politica consente `DescribeSecret` l'accesso ai segreti con un tag con la chiave *`ServerName`* e il valore*`ServerABC`*. Se alleghi questa politica a un'identità, l'identità è autorizzata a utilizzare tutti i segreti con quel tag nell'account.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
  "Effect": "Allow",
  "Action": "secretsmanager:DescribeSecret",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "secretsmanager:ResourceTag/ServerName": "ServerABC"
      }
    }
  }
}
```

Esempio: consenti l'accesso solo alle identità con tag che corrispondono ai tag dei segreti

La seguente politica consente a tutte le identità dell'account di `GetSecretValue` accedere a tutti i segreti dell'account in cui il *`AccessProject`* tag di identità ha lo stesso valore del tag del segreto. *`AccessProject`*

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
  "Effect": "Allow",
  "Principal": {
    "AWS": "123456789012"
  },
  "Condition": {
```

```
    "StringEquals": {
      "aws:ResourceTag/AccessProject": "${ aws:PrincipalTag/AccessProject }"
    }
  },
  "Action": "secretsmanager:GetSecretValue",
  "Resource": "*"
  }
}
```

AWS politica gestita per Gestione dei segreti AWS

Una politica AWS gestita è una politica autonoma creata e amministrata da AWS. AWS le politiche gestite sono progettate per fornire autorizzazioni per molti casi d'uso comuni, in modo da poter iniziare ad assegnare autorizzazioni a utenti, gruppi e ruoli.

Tieni presente che le policy AWS gestite potrebbero non concedere le autorizzazioni con il privilegio minimo per i tuoi casi d'uso specifici, poiché sono disponibili per tutti i clienti. AWS Si consiglia pertanto di ridurre ulteriormente le autorizzazioni definendo policy gestite dal cliente specifiche per i propri casi d'uso.

Non è possibile modificare le autorizzazioni definite nelle politiche gestite. AWS Se AWS aggiorna le autorizzazioni definite in una politica AWS gestita, l'aggiornamento ha effetto su tutte le identità principali (utenti, gruppi e ruoli) a cui è associata la politica. AWS è più probabile che aggiorni una policy AWS gestita quando ne Servizio AWS viene lanciata una nuova o quando diventano disponibili nuove operazioni API per i servizi esistenti.

Per ulteriori informazioni, consultare Policy gestite da AWS nella Guida per l'utente di IAM.

AWS politica gestita: SecretsManagerReadWrite

Questa policy fornisce read/write l'accesso Gestione dei segreti AWS, inclusa l'autorizzazione a descrivere, le risorse Amazon RDS, Amazon Redshift e Amazon DocumentDB e l'autorizzazione all' AWS KMS uso per crittografare e decrittografare i segreti. Questa policy consente inoltre di creare set di AWS CloudFormation modifiche, ottenere modelli di rotazione da un bucket Amazon S3 gestito da AWS, elencare AWS Lambda funzioni e descrivere Amazon EC2. VPCs Queste autorizzazioni sono richieste dalla console per impostare la rotazione con le funzioni di rotazione esistenti.

Per creare nuove funzioni di rotazione, devi inoltre disporre dell'autorizzazione a creare AWS CloudFormation stack e ruoli di esecuzione. AWS Lambda È possibile assegnare la politica gestita di IAMFullAccess. Per informazioni, consulta Autorizzazioni per la rotazione.

Dettagli delle autorizzazioni

Questa policy include le seguenti autorizzazioni:

- `secretsmanager`: consente ai principali di eseguire tutte le azioni di Secrets Manager.
- `cloudformation`— Consente ai principali di creare CloudFormation pile. Ciò è necessario affinché i principali che utilizzano la console per attivare la rotazione possano creare funzioni CloudFormation di rotazione Lambda tramite pile. Per ulteriori informazioni, consulta the section called "Come utilizza Secrets Manager CloudFormation".
- `ec2`— Consente ai responsabili di descrivere Amazon VPCs EC2. Ciò è necessario affinché i principali che utilizzano la console possano creare funzioni di rotazione nello stesso VPC del database delle credenziali che stanno archiviando in un segreto.
- `kms`— Consente ai responsabili di utilizzare le AWS KMS chiavi per le operazioni crittografiche. Ciò è necessario per consentire a Secrets Manager di crittografare e decrittografare i segreti. Per ulteriori informazioni, consulta the section called "Crittografia e decrittografia del segreto".
- `lambda`: consente ai principali di elencare le funzioni di rotazione Lambda. Ciò è necessario affinché i principali che utilizzano la console possano scegliere le funzioni di rotazione esistenti.
- `rds`: consente ai principali di descrivere i cluster e le istanze in Amazon RDS. Ciò è necessario affinché i principali che utilizzano la console possano scegliere cluster o istanze Amazon RDS.
- `redshift`: consente ai principali di descrivere i cluster in Amazon Redshift. Ciò è necessario affinché i principali che utilizzano la console possano scegliere cluster Amazon Redshift.
- `redshift-serverless`— Consente ai responsabili di descrivere i namespace in Amazon Redshift Serverless. Ciò è necessario affinché i responsabili che utilizzano la console possano scegliere i namespace Serverless di Amazon Redshift.
- `docdb-elastic`: consente ai principali di descrivere cluster elastici in Amazon DocumentDB. Ciò è necessario affinché i principali che utilizzano la console possano scegliere cluster elastici Amazon DocumentDB.
- `tag`: consente ai principali di ottenere tutte le risorse dell'account che sono contrassegnate.
- `serverlessrepo`— Consente ai principali di creare set di modifiche. CloudFormation Ciò è necessario affinché i principali che utilizzano la console possano creare funzioni di rotazione Lambda. Per ulteriori informazioni, consulta the section called "Come utilizza Secrets Manager CloudFormation".
- `s3`— Consente ai principali di ottenere oggetti da un bucket Amazon S3 gestito da. AWS Questo bucket contiene Lambda Modelli di funzione di rotazione. Questo permesso è necessario affinché i

principali che utilizzano la console possano creare funzioni di rotazione Lambda basate sui modelli nel bucket. Per ulteriori informazioni, consulta the section called "Come utilizza Secrets Manager CloudFormation".

Per visualizzare la policy, consulta il documento sulla policy SecretsManagerReadWrite JSON.

AWS politica gestita: AWSSecrets ManagerClientReadOnlyAccess

Questa policy fornisce l'accesso in sola lettura ai Gestione dei segreti AWS segreti per le applicazioni client. Consente ai responsabili di recuperare valori segreti e descrivere metadati segreti, oltre alle AWS KMS autorizzazioni necessarie per decrittografare i segreti crittografati con chiavi gestite dal cliente.

Dettagli delle autorizzazioni

Questa policy include le seguenti autorizzazioni:

- `secretsmanager`— Consente ai mandanti di recuperare valori segreti e descrivere metadati segreti.
- `kms`— Consente ai responsabili di decrittografare i segreti utilizzando le chiavi. AWS KMS Questa autorizzazione è limitata alle chiavi utilizzate da Secrets Manager in base a condizioni specifiche del servizio.

Per visualizzare ulteriori dettagli sulla policy, inclusa la versione più recente del documento di policy JSON, consulta AWSSecretsManagerClientReadOnlyAccess nella Guida di riferimento alle policy gestite da AWS .

Secrets Manager: aggiornamenti alle policy AWS gestite

Visualizza i dettagli sugli aggiornamenti delle politiche AWS gestite per Secrets Manager.

Modifica	Descrizione	Data	Versione
AWSSecretsManagerClientReadOnlyAccess— Nuova politica gestita	Secrets Manager ha creato una nuova policy gestita per fornire l'accesso in sola lettura ai	5 novembre 2025	v1

Modifica	Descrizione	Data	Versione
	segreti per le applicazioni client. Questa politica consente di recuperare valori segreti e descriver e metadati segreti, con le autorizzazioni necessarie per decrittografare i segreti. AWS KMS		
SecretsManagerReadWrite: aggiornamento di una policy esistente	Questa policy è stata aggiornata per consentire l'accesso descrittivo ad Amazon Redshift Serverless in modo che gli utenti della console possano scegliere uno spazio dei nomi Amazon Redshift Serverless quando creano un segreto Amazon Redshift.	12 marzo 2024	v5

Modifica	Descrizione	Data	Versione
SecretsManagerReadWrite: aggiornamento di una policy esistente	Questa policy è stata aggiornata per consentire l'accesso descrittivo ai cluster elastici di Amazon DocumentDB in modo che gli utenti della console possano scegliere un cluster elastico quando creano un segreto Amazon DocumentDB.	12 settembre 2023	v4

Modifica	Descrizione	Data	Versione
SecretsManagerReadWrite: aggiornamento di una policy esistente	Questa policy è stata aggiornata per consentire l'accesso descrittivo ad Amazon Redshift in modo che gli utenti della console possano scegliere un cluster Amazon Redshift quando creano un segreto Amazon Redshift. L'aggiornamento ha inoltre aggiunto nuove autorizzazioni per consentire l'accesso in lettura a un bucket Amazon S3 gestito AWS da che memorizza i modelli delle funzioni di rotazione Lambda.	24 giugno 2020	v3
SecretsManagerReadWrite: aggiornamento di una policy esistente	Questa policy è stata aggiornata per consentire l'accesso descrittivo ai cluster Amazon RDS in modo che gli utenti della console possano scegliere un cluster quando creano un segreto Amazon RDS.	03 maggio 2018	v2

Modifica	Descrizione	Data	Versione
SecretsManagerReadWrite: nuova policy	Secrets Manager ha creato una politica per concedere le autorizzazioni necessarie per utilizzare la console con tutti gli read/write accessi a Secrets Manager.	04 Aprile 2018	v1

Determina chi dispone delle autorizzazioni per accedere ai tuoi segreti Gestione dei segreti AWS

Per impostazione predefinita, le identità IAM non dispongono dell'autorizzazione per accedere ai segreti. Quando si autorizza l'accesso a un segreto, Secret Manager valuta la policy basata su risorse collegata al segreto e tutte le policy basato sull'identità collegate all'utente IAM o al ruolo da cui proviene la richiesta. Per eseguire questa operazione, Secret Manager utilizza un processo simile a quello descritto in Determinare se una richiesta è consentita o rifiutata nella Guida per l'utente IAM.

Quando a una richiesta si applicano varie policy, Gestione dei segreti utilizza una gerarchia per controllare le autorizzazioni:

1. Se una dichiarazione in qualsiasi politica con un esplicito `deny` corrisponde all'azione della richiesta e alla risorsa:

   L'esplicito `deny` sovrascrive tutto il resto e blocca l'azione.
2. Se non c'è esplicito `deny`, ma una dichiarazione con un esplicito `allow` corrisponde all'azione della richiesta e alla risorsa:

   L'esplicito `allow` concede l'azione nella richiesta di accesso alle risorse nell'istruzione.

   Se l'identità e il segreto si trovano in due account diversi, deve esserci una `allow` politica delle risorse per il segreto e una politica allegata all'identità, altrimenti la richiesta verrà AWS respinta. Per ulteriori informazioni, consulta Accesso multi-account.

3. Se non vi è alcuna dichiarazione con un esplicito `allow` che corrisponde all'azione di richiesta e alla risorsa:

   AWS nega la richiesta per impostazione predefinita, operazione denominata negazione implicita.

Per visualizzare le policy basate sulle risorse per un segreto

- Esegui una delle seguenti operazioni:
  - Apri la console Secrets Manager all'indirizzo https://console.aws.amazon.com/secretsmanager/. Nella pagina dei dettagli segreti per il segreto, nella sezione Autorizzazioni risorse, sceglieModifica autorizzazioni.
  - Usa il comando AWS CLI per chiamare `get-resource-policy`o l' AWS SDK per chiamare `GetResourcePolicy`.

Per determinare chi ha accesso tramite policy basate sull'identità

- Usa il simulatore di policy IAM. Vedere Test delle policy &IAM; con il simulatore di policy &IAM

Accedi ai Gestione dei segreti AWS segreti da un altro account

Per consentire agli utenti in un account di accedere ai segreti in un altro account (accesso tra account), è necessario consentire l'accesso sia in una policy delle risorse che in una policy di identità. Ciò è diverso dalla concessione dell'accesso alle identità nello stesso account del segreto.

L'autorizzazione tra account è valida solo per le seguenti operazioni:

- CancelRotateSecret
- DeleteResourcePolicy
- DeleteSecret
- DescribeSecret
- GetRandomPassword
- GetResourcePolicy
- GetSecretValue
- ListSecretVersionIds
- PutResourcePolicy

- PutSecretValue
- RemoveRegionsFromReplication
- ReplicateSecretToRegions
- RestoreSecret
- RotateSecret
- StopReplicationToReplica
- TagResource
- UntagResource
- UpdateSecret
- UpdateSecretVersionStage
- ValidateResourcePolicy

Puoi utilizzare il `BlockPublicPolicy` parametro insieme all'PutResourcePolicyazione per proteggere le tue risorse impedendo che l'accesso pubblico venga concesso tramite le politiche relative alle risorse direttamente collegate ai tuoi segreti. Puoi anche utilizzare IAM Access Analyzer per verificare l'accesso tra account.

È inoltre necessario consentire all'identità di utilizzare la chiave KMS con cui il segreto è crittografato. Questo perché non puoi usare il Chiave gestita da AWS (`aws/secretsmanager`) per l'accesso tra account diversi. È invece necessario crittografare il segreto con una chiave KMS creata e quindi allegare ad esso un criterio chiave. È previsto un addebito per la creazione di chiavi KMS. Per modificare la chiave di crittografia per un segreto, vedere the section called "Modificare un segreto".

> **Important**
>
> Le politiche basate sulle risorse che concedono `secretsmanager:PutResourcePolicy` l'autorizzazione offrono ai responsabili, anche a quelli di altri account, la possibilità di modificare le politiche basate sulle risorse. Questa autorizzazione consente ai responsabili di aumentare le autorizzazioni esistenti, ad esempio ottenere l'accesso amministrativo completo ai segreti. Ti consigliamo di applicare il principio dell'accesso minimo con privilegi alle tue politiche. Per ulteriori informazioni, consulta Policy basate sulle risorse.

I criteri di esempio seguenti presuppongono di disporre di una chiave segreta e di crittografia in Account1, e un'identità in Account2 che vuoi consentire ad accedere al valore segreto.

Fase 1: Allegare una policy delle risorse al segreto in Account1

- La seguente politica consente *ApplicationRole* a In *Account2* di accedere al secret in*Account1*. Per utilizzare questa policy, consultare the section called "Policy basate sulle risorse".

  JSON

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

Fase 2: Aggiungere un'istruzione alla policy della chiave per la chiave KMS in Account1

- La seguente dichiarazione sulla politica chiave consente *ApplicationRole* di *Account2* utilizzare la chiave KMS *Account1* per decrittografare il segreto in. *Account1* Per utilizzare questa istruzione, aggiungerla al criterio chiave per la chiave KMS. Per ulteriori informazioni, vedere Modifica di una policy delle chiavi.

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::Account2:role/ApplicationRole"
  },
  "Action": [
    "kms:Decrypt",
    "kms:DescribeKey"
  ],
  "Resource": "*"
```

```
}
```

Fase 3: Allegare una policy di identità all'identità in Account2

- La seguente politica consente di *ApplicationRole* accedere *Account2* al secret in *Account1* e decrittografare il valore segreto utilizzando la chiave di crittografia anch'essa presente in. *Account1* Per utilizzare questa policy, vedere the section called "Policy basate sull'identità". Puoi trovare l'ARN per il tuo segreto nella console di Secrets Manager nella pagina dei dettagli segreti sotto ARN del segreto. In alternativa, è possibile chiamare `describe-secret`.

  JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "secretsmanager:GetSecretValue",
            "Resource": "arn:aws:secretsmanager:us-
east-1:123456789012:secret:secretName-AbCdEf"
        },
        {
            "Effect": "Allow",
            "Action": "kms:Decrypt",
            "Resource": "arn:aws:kms:us-
east-1:123456789012:key/EncryptionKey"
        }
    ]
}
```

Accedi ai segreti da un ambiente locale

Puoi utilizzare AWS Identity and Access Management Roles Anywhere per ottenere credenziali di sicurezza temporanee in IAM per carichi di lavoro come server, contenitori e applicazioni che vengono eseguiti all'esterno di. AWS I tuoi carichi di lavoro possono utilizzare le stesse politiche IAM e gli stessi ruoli IAM che utilizzi con AWS le applicazioni per accedere alle risorse. AWS Con IAM Roles Anywhere, puoi utilizzare Secrets Manager per archiviare e gestire le credenziali alle quali

possono accedere sia le risorse in AWS sia i dispositivi on-premise, come i server delle applicazioni. Per ulteriori informazioni, consulta la Guida per l'utente di IAM Roles Anywhere.

Protezione dei dati in Gestione dei segreti AWS

Il modello di responsabilità AWS condivisa di si applica alla protezione dei dati in Gestione dei segreti AWS. Come descritto in questo modello, AWS è responsabile della protezione dell'infrastruttura globale che gestisce tutti i Cloud AWS. L'utente è responsabile di mantenere il controllo sui contenuti ospitati su questa infrastruttura. Questi contenuti comprendono la configurazione della protezione e le attività di gestione per i Servizi AWS utilizzati. Per ulteriori informazioni sulla privacy dei dati, vedi le Domande frequenti sulla privacy dei dati. Per informazioni sulla protezione dei dati in Europa, consulta il post del blog relativo al AWS Modello di responsabilità condivisa e GDPR nel AWS Blog sulla sicurezza.

Ai fini della protezione dei dati, consigliamo di proteggere Account AWS le credenziali e configurare account utente individuali con AWS Identity and Access Management (IAM). In questo modo, a ogni utente verranno assegnate solo le autorizzazioni necessarie per svolgere il proprio lavoro. Ti suggeriamo, inoltre, di proteggere i dati nei seguenti modi:

- Utilizza l'autenticazione a più fattori (MFA) con ogni account.
- SSL/TLS Da utilizzare per comunicare con AWS le risorse. Secrets Manager supporta TLS 1.2 e 1.3 in tutte le Regioni. Secrets Manager supporta anche un'opzione ibrida di scambio di chiavi post-quantistiche per il protocollo di crittografia di rete TLS (PQTLS).
- Firma le tue richieste programmatiche utilizzando sia un ID chiave di accesso che una chiave di accesso segreta associate a un'entità principale IAM. In alternativa, è possibile utilizzare AWS Security Token Service (AWS STS) per generare le credenziali di sicurezza temporanee per firmare le richieste.
- Configura l'API e la registrazione delle attività degli utenti con AWS CloudTrail. Per informazioni, consulta the section called “Accedi con AWS CloudTrail ”.
- Se hai bisogno di moduli crittografici convalidati FIPS 140-2 per l'accesso AWS tramite un'interfaccia a riga di comando o un'API, utilizza un endpoint FIPS. Per informazioni, consulta the section called “Endpoint di Secrets Manager”.
- Se usi il AWS CLI per accedere a Secrets Manager,the section called “Riduci i rischi derivanti dall'utilizzo di per archiviare i tuoi AWS CLI segreti Gestione dei segreti AWS”.

Crittografia dei dati a riposo

Secrets Manager utilizza la crittografia tramite AWS Key Management Service (AWS KMS) per proteggere la riservatezza dei dati archiviati. AWS KMS fornisce un servizio di archiviazione e crittografia delle chiavi utilizzato da molti AWS servizi. Ogni segreto in Gestione dei segreti è crittografato con una chiave di dati univoca. Ogni chiave di dati è protetta da una chiave KMS. Puoi scegliere di utilizzare la crittografia predefinita con la Chiave gestita da AWS di Gestione dei segreti per l'account. In alternativa, puoi creare la tua chiave gestita dal cliente in AWS KMS. L'utilizzo di una chiave gestita dal cliente offre controlli di autorizzazione più granulari sulle principali attività del KMS. Per ulteriori informazioni, consulta the section called “Crittografia e decrittografia del segreto”.

Crittografia dei dati in transito

Secrets Manager fornisce endpoint sicuri e privati per la crittografia dei dati in transito. Gli endpoint sicuri e privati consentono di AWS proteggere l'integrità delle richieste API a Secrets Manager. AWS richiede che le chiamate API siano firmate dal chiamante utilizzando certificati X.509 e una chiave di accesso segreta di Secrets and/or Manager. Questo requisito è indicato nel Processo di firma Signature Version 4 (Sigv4).

Se si utilizza il AWS Command Line Interface (AWS CLI) o uno dei seguenti AWS SDKs per effettuare chiamate AWS, si configura la chiave di accesso da utilizzare. Quindi questi strumenti utilizzano automaticamente la chiave di accesso per firmare le richieste per te. Per informazioni, consulta the section called “Riduci i rischi derivanti dall'utilizzo di per archiviare i tuoi AWS CLI segreti Gestione dei segreti AWS”.

Riservatezza del traffico inter-rete

AWS offre opzioni per mantenere la privacy durante l'instradamento del traffico attraverso percorsi di rete noti e privati.

Traffico tra servizio e applicazioni e client locali

Sono disponibili due opzioni di connettività tra la rete privata e Gestione dei segreti AWS:

- Una connessione AWS Site-to-Site VPN. Per ulteriori informazioni, consulta Cos'è la AWS VPN da sito a sito?
- Una connessione AWS Direct Connect. Per ulteriori informazioni, consulta Che cos'è AWS Direct Connect?

Traffico tra AWS risorse nella stessa regione

Se desideri proteggere il traffico tra Secrets Manager e i client API in AWS, configura un endpoint dell'API Secrets Manager AWS PrivateLinkper accedere in modo privato.

Gestione delle chiavi crittografiche

Quando Secrets Manager deve crittografare una nuova versione dei dati segreti protetti, Secrets Manager invia una richiesta AWS KMS per generare una nuova chiave dati dalla chiave KMS. Secrets Manager utilizza questa chiave di dati per la crittografia envelope. Secrets Manager archivia la chiave di dati crittografati con il segreto crittografato. Quando il segreto deve essere decrittografato, Secrets Manager chiede di AWS KMS decrittografare la chiave dati. Quindi Secrets Manager usa la chiave di dati decrittografata per decrittografare il segreto crittografato. La chiave di dati non viene mai memorizzata in forma non crittografata da Secrets Manager e la chiave viene rimossa dalla memoria il prima possibile. Per ulteriori informazioni, consulta the section called "Crittografia e decrittografia del segreto".

Crittografia e decrittografia segrete in Gestione dei segreti AWS

Secrets Manager utilizza la crittografia a busta con AWS KMS chiavi e chiavi dati per proteggere ogni valore segreto. Ogni volta che il valore del segreto in un segreto cambia, Secrets Manager richiede una nuova chiave da AWS KMS per proteggerlo. La chiave di dati è crittografata sotto una chiave KMS e viene archiviata nei metadati della versione segreta. Per decrittografare il segreto, Secrets Manager decrittografa innanzitutto la chiave dati crittografata utilizzando la chiave KMS. AWS KMS

Secrets Manager non utilizza la chiave KMS per crittografare il valore del segreto direttamente. Utilizza invece la chiave KMS per generare e crittografare una simmetrica AES (Advanced Encryption Standard) a 256 bit chiave di dati e utilizza la chiave di dati per crittografare il valore del segreto. Secrets Manager utilizza la chiave dati in chiaro per crittografare il valore segreto all'esterno di AWS KMS, quindi lo rimuove dalla memoria. Archivia la copia crittografata della chiave di dati nei metadati del segreto.

Argomenti

- Come Secrets Manager utilizza la chiave KMS
- Politica chiave di Chiave gestita da AWS () aws/secretsmanager
- Contesto di crittografia di Secrets Manager
- Monitora l'interazione di Secrets Manager con AWS KMS

Scelta di una chiave AWS KMS

Quando crei un segreto, puoi scegliere qualsiasi chiave di crittografia simmetrica gestita dal cliente nella regione Account AWS and oppure puoi utilizzare Chiave gestita da AWS for Secrets Manager ()`aws/secretsmanager`. Se scegli il Chiave gestita da AWS `aws/secretsmanager` e non esiste ancora, Secrets Manager lo crea e lo associa al segreto. È possibile utilizzare la stessa chiave KMS o diverse chiavi KMS per ogni segreto nell'account. Potresti voler utilizzare chiavi KMS diverse per impostare autorizzazioni personalizzate sulle chiavi per un gruppo di segreti o per controllare determinate operazioni per tali chiavi. Secrets Manager supporta solo chiavi KMS di crittografia simmetrica. Se utilizzi una chiave KMS in un archivio di chiavi esterno, le operazioni crittografiche sulla chiave KMS potrebbero richiedere più tempo ed essere meno affidabili e durevoli perché la richiesta deve essere trasferita all'esterno di AWS.

Per informazioni sulla modifica della chiave di crittografia di un segreto, consulta the section called "Modifica la chiave di crittografia per un segreto".

Quando si modifica la chiave di crittografia, Secrets Manager cripta nuovamente `AWSCURRENT` e `AWSPENDING` le `AWSPREVIOUS` versioni con la nuova chiave. Per evitare di nasconderti il segreto, Secrets Manager mantiene tutte le versioni esistenti crittografate con la chiave precedente. Ciò significa che è possibile decrittografare `AWSCURRENT` `AWSPREVIOUS` le versioni con la chiave precedente o la nuova chiave. `AWSPENDING` Se non si dispone dell'`kms:Decrypt`autorizzazione per la chiave precedente, quando si modifica la chiave di crittografia, Secrets Manager non può decrittografare le versioni segrete per crittografarle nuovamente. In questo caso, le versioni esistenti non vengono ricrittografate.

Per fare in modo che `AWSCURRENT` possa essere decifrata solo con la nuova chiave di crittografia, crea una nuova versione del segreto con la nuova chiave. Quindi, per poter decifrare la versione `AWSCURRENT` segreta, devi avere l'autorizzazione per la nuova chiave.

È possibile negare l'autorizzazione Chiave gestita da AWS `aws/secretsmanager` e richiedere che i segreti vengano crittografati con una chiave gestita dal cliente. Per ulteriori informazioni, consulta the section called "Esempio: nega una AWS KMS chiave specifica per crittografare i segreti".

Per trovare la chiave KMS associata a un segreto, visualizza il segreto nella console o chiama ListSecretso. DescribeSecret Quando il segreto è associato a Chiave gestita da AWS for Secrets Manager (`aws/secretsmanager`), queste operazioni non restituiscono un identificatore di chiave KMS.

Che viene crittografato?

Secrets Manager crittografa il valore segreto, ma non crittografa quanto segue:

- Nome e descrizione del segreto
- Impostazioni di rotazione
- ARN della chiave KMS associata al segreto
- Eventuali tag allegati AWS

Processi di crittografia e decrittografia

Per crittografare il valore del segreto in un segreto, Secrets Manager utilizza il seguente processo.

1. Secrets Manager richiama l' AWS KMS GenerateDataKeyoperazione con l'ID della chiave KMS per il segreto e una richiesta per una chiave simmetrica AES a 256 bit. AWS KMS restituisce una chiave di dati in testo semplice e una copia di tale chiave dati crittografata con la chiave KMS.
2. Secrets Manager utilizza la chiave dati in chiaro e l'algoritmo Advanced Encryption Standard (AES) per crittografare il valore segreto all'esterno di. AWS KMS Dopo averla utilizzata, rimuove la chiave in testo normale dalla memoria il prima possibile.
3. Secrets Manager archivia la chiave di dati crittografata nei metadati del segreto in modo che sia disponibile per decrittografare il valore del segreto. Tuttavia, nessuno dei Secrets Manager APIs restituisce il segreto crittografato o la chiave dati crittografata.

Per decrittografare un valore del segreto crittografato:

1. Secrets Manager richiama l'operazione AWS KMS Decrypt e trasmette la chiave dati crittografata.
2. AWS KMS utilizza la chiave KMS come segreto per decrittografare la chiave dati. Restituisce la chiave di dati in testo normale.
3. Secrets Manager utilizza la chiave di dati in testo normale per decrittografare il valore del segreto. Quindi rimuove la chiave di dati dalla memoria il prima possibile.

Autorizzazioni per la chiave KMS

Quando Secrets Manager utilizza una chiave KMS in operazioni di crittografia, agisce per conto dell'utente che sta effettuando l'accesso al valore del segreto o che lo sta aggiornando. Puoi concedere le autorizzazioni in una policy IAM o in una policy delle chiavi. Le seguenti operazioni di Secrets Manager richiedono AWS KMS autorizzazioni.

- CreateSecret
- GetSecretValue
- PutSecretValue
- UpdateSecret
- ReplicateSecretToRegions

Per consentire l'utilizzo della chiave KMS solo per le richieste che hanno origine in Secrets Manager, nella politica delle autorizzazioni, puoi utilizzare la chiave kms: ViaService condition con il valore. `secretsmanager.<Region>.amazonaws.com`

Puoi inoltre utilizzare le chiavi o i valori nel contesto di crittografia come condizione per utilizzare la chiave KMS per le operazioni di crittografia. Ad esempio, è possibile utilizzare un operatore di condizione stringa in un documento di policy IAM o della chiave, oppure utilizzare un vincolo di concessione in un vincolo. La propagazione della concessione di chiavi KMS può richiedere fino a cinque minuti. Per ulteriori informazioni, consulta CreateGrant.

Come Secrets Manager utilizza la chiave KMS

Secrets Manager richiama le seguenti AWS KMS operazioni con la chiave KMS.

GenerateDataKey

Secrets Manager chiama l' AWS KMS GenerateDataKeyoperazione in risposta alle seguenti operazioni di Secrets Manager.

- CreateSecret— Se il nuovo segreto include un valore segreto, Secrets Manager richiede una nuova chiave dati per crittografarlo.
- PutSecretValue— Secrets Manager richiede una nuova chiave dati per crittografare il valore segreto specificato.
- ReplicateSecretToRegions— Per crittografare il segreto replicato, Secrets Manager richiede una chiave dati per la chiave KMS nella regione di replica.

- UpdateSecret— Se si modifica il valore segreto o la chiave KMS, Secrets Manager richiede una nuova chiave dati per crittografare il nuovo valore segreto.

L'RotateSecretoperazione non chiamaGenerateDataKey, perché non modifica il valore segreto. Tuttavia, se RotateSecret invoca una funzione di rotazione Lambda che modifica il valore del segreto, la chiamata all'operazione PutSecretValue attiva una richiesta GenerateDataKey.

Decrypt

Secrets Manager chiama la Decrypt operazione in risposta alle seguenti operazioni di Secrets Manager.

- GetSecretValuee BatchGetSecretValue— Secrets Manager decripta il valore segreto prima di restituirlo al chiamante. Per decrittografare un valore segreto crittografato, Secrets Manager chiama l'operazione AWS KMS Decrypt per decrittografare la chiave dei dati crittografati nel segreto. Quindi utilizza la chiave di dati in testo normale per decrittografare il valore del segreto crittografato. Per i comandi batch, Secrets Manager può riutilizzare la chiave decrittografata, in modo che non tutte le chiamate generino una richiesta Decrypt.
- PutSecretValuee UpdateSecret— La maggior parte delle UpdateSecret richieste non PutSecretValue attiva un'operazione. Decrypt Tuttavia, quando una richiesta PutSecretValue o UpdateSecret cerca di modificare il valore del segreto in una versione esistente di un segreto, Secrets Manager decrittografa il valore del segreto esistente e lo confronta con il valore del segreto nella richiesta per confermare che siano identici. Questa operazione garantisce che le operazioni Secrets Manager siano idempotenti. Per decrittografare un valore segreto crittografato, Secrets Manager chiama l'operazione AWS KMS Decrypt per decrittografare la chiave dei dati crittografati nel segreto. Quindi utilizza la chiave di dati in testo normale per decrittografare il valore del segreto crittografato.
- ReplicateSecretToRegions— Secrets Manager decrittografa innanzitutto il valore segreto nella regione primaria prima di ricrittografare il valore segreto con la chiave KMS nella regione di replica.

Crittografa

Secrets Manager chiama l'operazione Encrypt in risposta alle seguenti operazioni di Secrets Manager:

- UpdateSecret— Se si modifica la chiave KMS, Secrets Manager cripta nuovamente la chiave dati che protegge la AWSCURRENT chiave e le versioni AWSPENDING segrete con la nuova chiave. AWSPREVIOUS

DescribeKey

Secrets Manager richiama l'DescribeKeyoperazione per determinare se elencare la chiave KMS quando si crea o si modifica un segreto nella console di Secrets Manager.

Convalida dell'accesso alla chiave KMS

Quando stabilisci o modifichi la chiave KMS associata al segreto, Secrets Manager chiama le operazioni `GenerateDataKey` e `Decrypt` con la chiave KMS specificata. Queste chiamate confermano che il chiamante ha l'autorizzazione di utilizzare la chiave KMS per queste operazioni. Secrets Manager scarta i risultati di queste operazioni e non li utilizza in alcuna operazione di crittografia.

È possibile identificare queste chiamate di convalida perché il valore della `SecretVersionId` chiave contesto di crittografia in queste richieste è `RequestToValidateKeyAccess`.

> **Note**
>
> In passato, le chiamate di convalida di Secrets Manager non includevano un contesto di crittografia. È possibile trovare chiamate senza contesto di crittografia nei AWS CloudTrail registri più vecchi.

Politica chiave di Chiave gestita da AWS () **`aws/secretsmanager`**

La politica chiave per Chiave gestita da AWS for Secrets Manager (`aws/secretsmanager`) consente agli utenti di utilizzare la chiave KMS per operazioni specifiche solo quando Secrets Manager effettua la richiesta per conto dell'utente. La policy delle chiavi non consente ad alcun utente di utilizzare la chiave KMS direttamente.

Questa policy delle chiavi, come le policy di tutte le Chiavi gestite da AWS, viene stabilita dal servizio. Non è possibile modificarla, ma è possibile visualizzarla in qualsiasi momento. Per informazioni dettagliate, consulta Visualizzazione di una policy di chiave.

Le istruzioni di policy nella policy delle chiavi hanno l'effetto seguente:

- Consenti agli utenti nell'account di utilizzare la chiave KMS per le operazioni di crittografia solo quando la richiesta proviene da Secrets Manager per conto loro. La chiave di condizione `kms:ViaService` applica questa limitazione.

- Consente all' AWS account di creare policy IAM che consentono agli utenti di visualizzare le proprietà delle chiavi KMS e revocare le concessioni.
- Sebbene Secrets Manager non utilizzi le sovvenzioni per accedere alla chiave KMS, la policy consente inoltre a Secrets Manager di creare sovvenzioni per la chiave KMS per conto dell'utente e consente all'account di revoca di qualsiasi concessione che consente a Secrets Manager di utilizzare la chiave KMS. Questi sono elementi standard del documento politico per un. Chiave gestita da AWS

Di seguito è riportata una politica chiave per un Chiave gestita da AWS esempio di Secrets Manager.

JSON

```
{
  "Id": "auto-secretsmanager-2",
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "Allow access through AWS Secrets Manager for all principals in the
 account that are authorized to use AWS Secrets Manager",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "*"
        ]
      },
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:CreateGrant",
        "kms:DescribeKey"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "kms:CallerAccount": "111122223333",
          "kms:ViaService": "secretsmanager.us-west-2.amazonaws.com"
        }
      }
    },
    {
```

```
      "Sid": "Allow access through AWS Secrets Manager for all principals in the
  account that are authorized to use AWS Secrets Manager",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "*"
        ]
      },
      "Action": "kms:GenerateDataKey*",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "kms:CallerAccount": "111122223333"
        },
        "StringLike": {
          "kms:ViaService": "secretsmanager.us-west-2.amazonaws.com"
        }
      }
    },
    {
      "Sid": "Allow direct access to key metadata to the account",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::111122223333:root"
        ]
      },
      "Action": [
        "kms:Describe*",
        "kms:Get*",
        "kms:List*",
        "kms:RevokeGrant"
      ],
      "Resource": "*"
    }
  ]
}
```

Contesto di crittografia di Secrets Manager

Un contesto di crittografia è un insieme di coppie chiave-valore che contengono dati arbitrari non segreti. Quando includi un contesto di crittografia in una richiesta di crittografia dei dati, associa AWS

KMS crittograficamente il contesto di crittografia ai dati crittografati. lo stesso contesto di crittografia sia necessario per decrittografare i dati.

Nelle sue richieste GenerateDataKeye Decrypt a AWS KMS, Secrets Manager utilizza un contesto di crittografia con due coppie nome-valore che identificano il segreto e la relativa versione, come illustrato nell'esempio seguente. I nomi non variano, ma i valori del contesto di crittografia combinati saranno diversi per ogni valore del segreto.

```
"encryptionContext": {
    "SecretARN": "arn:aws:secretsmanager:us-east-2:111122223333:secret:test-secret-
a1b2c3",
    "SecretVersionId": "EXAMPLE1-90ab-cdef-fedc-ba987SECRET1"
}
```

Puoi utilizzare il contesto di crittografia per identificare queste operazioni crittografiche nei record e nei log di controllo, come AWS CloudTrailAmazon CloudWatch Logs, e come condizione per l'autorizzazione nelle politiche e nelle concessioni.

Il contesto di crittografia di Secrets Manager è costituito da due coppie nome-valore.

- SecretArn – La coppia nome-valore identificherà il segreto. La chiave è `SecretARN`. Il valore è l'Amazon Resource Name (ARN) del segreto.

  ```
  "SecretARN": "ARN of an Secrets Manager secret"
  ```

  Ad esempio, se l'ARN del segreto è `arn:aws:secretsmanager:us-east-2:111122223333:secret:test-secret-a1b2c3`, il contesto di crittografia include la seguente coppia.

  ```
  "SecretARN": "arn:aws:secretsmanager:us-east-2:111122223333:secret:test-secret-
  a1b2c3"
  ```

- SecretVersionId— La seconda coppia nome-valore identifica la versione del segreto. La chiave è `SecretVersionId`. Il valore è l'ID della versione.

  ```
  "SecretVersionId": "<version-id>"
  ```

  Ad esempio, se l'ID della versione è `EXAMPLE1-90ab-cdef-fedc-ba987SECRET1`, il contesto di crittografia include la seguente coppia.

```
"SecretVersionId": "EXAMPLE1-90ab-cdef-fedc-ba987SECRET1"
```

Quando stabilisci o modifichi la chiave KMS per un segreto, Secrets Manager invia GenerateDataKeye AWS KMS decrittografa le richieste per verificare che il chiamante sia autorizzato a utilizzare la chiave KMS per queste operazioni. Scarta le risposte, non le utilizza sul valore del segreto.

In queste richieste di convalida, il valore di `SecretARN` è l'ARN effettivo del segreto, ma il valore `SecretVersionId` è `RequestToValidateKeyAccess`, come visualizzato nel seguente contesto di crittografia di esempio. Questo valore speciale consente di identificare le richieste di convalida nei log e negli audit trail.

```
"encryptionContext": {
    "SecretARN": "arn:aws:secretsmanager:us-east-2:111122223333:secret:test-secret-
a1b2c3",
    "SecretVersionId": "RequestToValidateKeyAccess"
}
```

> **Note**
>
> In passato, le richieste di convalida di Secrets Manager non includevano un contesto di crittografia. È possibile trovare chiamate prive di contesto di crittografia nei registri più vecchi. AWS CloudTrail

Monitora l'interazione di Secrets Manager con AWS KMS

Puoi utilizzare AWS CloudTrail Amazon CloudWatch Logs per tenere traccia delle richieste a cui Secrets Manager invia per tuo AWS KMS conto. Per ulteriori informazioni sul monitoraggio dell'uso dei segreti , consulta Monitorare i segreti.

GenerateDataKey

Quando crei o modifichi il valore segreto in un segreto, Secrets Manager invia una GenerateDataKeyrichiesta a AWS KMS cui specifica la chiave KMS per il segreto.

L'evento che registra l'operazione `GenerateDataKey` è simile a quello del seguente evento di esempio. La richiesta viene richiamata da `secretsmanager.amazonaws.com`. I parametri

includono l'Amazon Resource Name (ARN) della chiave KMS per il segreto, un identificatore della chiave che richiede una chiave a 256 bit e il contesto di crittografia che identifica il segreto e la versione.


```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AROAIGDTESTANDEXAMPLE:user01",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/user01",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2018-05-31T23:23:41Z"
            }
        },
        "invokedBy": "secretsmanager.amazonaws.com"
    },
    "eventTime": "2018-05-31T23:23:41Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateDataKey",
    "awsRegion": "us-east-2",
    "sourceIPAddress": "secretsmanager.amazonaws.com",
    "userAgent": "secretsmanager.amazonaws.com",
    "requestParameters": {
        "keyId": "arn:aws:kms:us-
east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "keySpec": "AES_256",
        "encryptionContext": {
            "SecretARN": "arn:aws:secretsmanager:us-east-2:111122223333:secret:test-
secret-a1b2c3",
            "SecretVersionId": "EXAMPLE1-90ab-cdef-fedc-ba987SECRET1"
        }
    },
    "responseElements": null,
    "requestID": "a7d4dd6f-6529-11e8-9881-67744a270888",
    "eventID": "af7476b6-62d7-42c2-bc02-5ce86c21ed36",
    "readOnly": true,
    "resources": [
        {
```

```
            "ARN": "arn:aws:kms:us-
east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
            "accountId": "111122223333",
            "type": "AWS::KMS::Key"
        }
    ],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
}
```

Decrypt

Quando si ottiene o si modifica il valore segreto di un segreto, Secrets Manager invia una richiesta di decrittografia per AWS KMS decrittografare la chiave dati crittografata. Per i comandi batch, Secrets Manager può riutilizzare la chiave decrittografata, in modo che non tutte le chiamate generino una richiesta `Decrypt`.

L'evento che registra l'operazione `Decrypt` è simile a quello del seguente evento di esempio. L'utente è il principale del tuo AWS account che accede alla tabella. I parametri includono la chiave crittografata della tabella (come blob di testo cifrato) e il contesto di crittografia che identifica la tabella e l'account. AWS AWS KMS ricava l'ID della chiave KMS dal testo cifrato.

```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AROAIGDTESTANDEXAMPLE:user01",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/user01",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2018-05-31T23:36:09Z"
            }
        },
        "invokedBy": "secretsmanager.amazonaws.com"
    },
    "eventTime": "2018-05-31T23:36:09Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "Decrypt",
    "awsRegion": "us-east-2",
    "sourceIPAddress": "secretsmanager.amazonaws.com",
```

```
    "userAgent": "secretsmanager.amazonaws.com",
    "requestParameters": {
        "encryptionContext": {
            "SecretARN": "arn:aws:secretsmanager:us-east-2:111122223333:secret:test-
secret-a1b2c3",
            "SecretVersionId": "EXAMPLE1-90ab-cdef-fedc-ba987SECRET1"
        }
    },
    "responseElements": null,
    "requestID": "658c6a08-652b-11e8-a6d4-ffee2046048a",
    "eventID": "f333ec5c-7fc1-46b1-b985-cbda13719611",
    "readOnly": true,
    "resources": [
        {
            "ARN": "arn:aws:kms:us-
east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
            "accountId": "111122223333",
            "type": "AWS::KMS::Key"
        }
    ],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
}
```

Crittografa

Quando si modifica la chiave KMS associata a un segreto, Secrets Manager invia una richiesta di crittografia per AWS KMS crittografare nuovamente le versioni `AWSPENDING` segrete con la `AWSCURRENT` nuova chiave. `AWSPREVIOUS` Quando replichi un segreto in un'altra regione, Secrets Manager invia anche una richiesta Encrypt a AWS KMS.

L'evento che registra l'operazione `Encrypt` è simile a quello del seguente evento di esempio. L'utente è il principale del tuo AWS account che accede alla tabella.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AROAIGDTESTANDEXAMPLE:user01",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/user01",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
```

```
            "attributes": {
                "creationDate": "2023-06-09T18:11:34Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "secretsmanager.amazonaws.com"
    },
    "eventTime": "2023-06-09T18:11:34Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "Encrypt",
    "awsRegion": "us-east-2",
    "sourceIPAddress": "secretsmanager.amazonaws.com",
    "userAgent": "secretsmanager.amazonaws.com",
    "requestParameters": {
        "keyId": "arn:aws:kms:us-east-2:111122223333:key/EXAMPLE1-f1c8-4dce-8777-
aa071ddefdcc",
        "encryptionAlgorithm": "SYMMETRIC_DEFAULT",
        "encryptionContext": {
            "SecretARN": "arn:aws:secretsmanager:us-
east-2:111122223333:secret:ChangeKeyTest-5yKnKS",
            "SecretVersionId": "EXAMPLE1-5c55-4d7c-9277-1b79a5e8bc50"
        }
    },
    "responseElements": null,
    "requestID": "129bd54c-1975-4c00-9b03-f79f90e61d60",
    "eventID": "f7d9ff39-15ab-47d8-b94c-56586de4ab68",
    "readOnly": true,
    "resources": [
        {
            "accountId": "AWS Internal",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:111122223333:key/EXAMPLE1-f1c8-4dce-8777-
aa071ddefdcc"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

Sicurezza dell'infrastruttura in Gestione dei segreti AWS

In quanto servizio gestito, Gestione dei segreti AWS è protetto dalla sicurezza della rete AWS globale. Per informazioni sui servizi AWS di sicurezza e su come AWS protegge l'infrastruttura, consulta AWS Cloud Security. Per progettare il tuo AWS ambiente utilizzando le migliori pratiche per la sicurezza dell'infrastruttura, vedi Infrastructure Protection in Security Pillar AWS Well-Architected Framework.

L'accesso a Secrets Manager tramite la rete avviene tramite AWS pubblicazione APIs tramite TLS. I Secrets Manager APIs sono richiamabili da qualsiasi posizione di rete. Tuttavia, Secrets Manager non supporta le policy di accesso basate sulle risorse, che possono includere limitazioni in base all'indirizzo IP di origine. Puoi anche utilizzare le policy delle risorse di Secrets Manager per controllare l'accesso ai segreti da endpoint VPC (Virtual Private Cloud) specifici o specifici. VPCs In effetti, questo isola l'accesso alla rete a un determinato segreto solo dal VPC specifico all'interno AWS della rete. Per ulteriori informazioni, consulta the section called “Endpoint VPC (AWS PrivateLink)”.

Utilizzo di un Gestione dei segreti AWS endpoint VPC

Consigliamo di eseguire la maggior parte delle infrastrutture su reti private non accessibili da Internet pubblico. È possibile stabilire una connessione privata tra il VPC e Secrets Manager creando un endpoint VPC dell'interfaccia. Gli endpoint di interfaccia sono alimentati da AWS PrivateLink, una tecnologia che consente di accedere in modo privato a Secrets Manager APIs senza un gateway Internet, un dispositivo NAT, una connessione VPN o una connessione. Direct Connect Le istanze nel tuo VPC non necessitano di indirizzi IP pubblici per comunicare con Secrets Manager. APIs Il traffico tra il tuo VPC e Secrets Manager non esce dalla AWS rete. Per ulteriori informazioni, consultare Endpoint VPC di interfaccia (AWS PrivateLink) nella Guida per l'utente di Amazon VPC.

Quando Secrets Manager effettua una rotazione del segreto utilizzando una funzione di rotazione Lambda, ad esempio un segreto che contiene credenziali del database, la funzione Lambda effettua richieste sia al database che a Secrets Manager. Quando si attiva la rotazione automatica utilizzando la console, Secrets Manager crea la funzione Lambda nello stesso VPC del database. Suggeriamo di creare un endpoint di Secrets Manager nello stesso VPC in modo che le richieste dalla funzione di rotazione Lambda a Secrets Manager non escano dalla rete Amazon.

Se si abilita il DNS privato per l'endpoint, è possibile effettuare richieste API verso Secrets Manager utilizzando il nome DNS predefinito per la regione, ad esempio `secretsmanager.us-`

`east-1.amazonaws.com`. Per ulteriori informazioni, consultare Accesso a un servizio tramite un endpoint di interfaccia in Guida per l'utente di Amazon VPC.

È possibile assicurarsi che le richieste a Secrets Manager provengano dall'accesso VPC includendo una condizione nelle policy di autorizzazione. Per ulteriori informazioni, consulta the section called "Esempio: autorizzazioni e VPCs".

Puoi utilizzare AWS CloudTrail i log per verificare l'utilizzo dei segreti tramite l'endpoint VPC.

Creare un endpoint VPC privato di Secrets Manager

1. Consulta Creazione di un endpoint di interfaccia nella Amazon VPC User Guide. Usa uno dei seguenti nomi di servizio:
   - `com.amazonaws.region.secretsmanager`
   - `com.amazonaws.region.secretsmanager-fips`
2. Per controllare l'accesso all'endpoint, consulta Controllare l'accesso agli endpoint VPC utilizzando le policy degli endpoint.
3. Per l'utilizzo di un indirizzamento IPv6 dual-stack, consulta. IPv4 e IPv6 accesso

Creazione di una policy dell' endpoint per l'endpoint dell'interfaccia

Una policy dell'endpoint è una risorsa IAM che è possibile allegare all'endpoint dell'interfaccia. La policy predefinita per gli endpoint consente l'accesso completo a Secrets Manager tramite l'endpoint dell'interfaccia. Per controllare l'accesso consentito a Secrets Manager dal tuo VPC, collega una policy personalizzata per gli endpoint all'endpoint dell'interfaccia.

Una policy di endpoint specifica le informazioni riportate di seguito:

- I principali che possono eseguire azioni (Account AWS, utenti IAM e ruoli IAM).
- Le azioni che possono essere eseguite.
- Le risorse in cui è possibile eseguire le operazioni.

Per ulteriori informazioni, consulta la sezione Controllo dell'accesso ai servizi con policy di endpoint nella Guida di AWS PrivateLink .

Esempio: policy degli endpoint VPC per le azioni di Secrets Manager

Di seguito è riportato l'esempio di una policy dell'endpoint personalizzata. Quando alleghi questa policy all'endpoint dell'interfaccia, concede l'accesso alle azioni di Secrets Manager elencate sul segreto specificato.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "Allow all users to use GetSecretValue and DescribeSecret on the
 specified secret.",
      "Effect": "Allow",
      "Principal": "*",
      "Action": [
        "secretsmanager:GetSecretValue",
        "secretsmanager:DescribeSecret"
      ],
      "Resource": "arn:aws:secretsmanager:us-
east-1:111122223333:secret:secretName-AbCdEf"
    }
  ]
}
```

Sottoreti condivise

Non puoi creare, descrivere, modificare o eliminare gli endpoint VPC nelle sottoreti condivise con te. Tuttavia, puoi utilizzare gli endpoint VPC in sottoreti condivise con te. Per informazioni sulla condivisione VPC, consulta Condivisione del VPC con altri account nella Guida per l'utente di Amazon Virtual Private Cloud.

Controlla l'accesso alle API con le policy IAM

Se utilizzi le policy IAM per controllare l'accesso in Servizi AWS base agli indirizzi IP, potresti dover aggiornare le policy per includere gli intervalli di IPv6 indirizzi. Questa guida spiega le differenze tra IPv4 IPv6 e descrive come aggiornare le policy IAM per supportare entrambi i protocolli. L'implementazione di queste modifiche consente di mantenere un accesso sicuro alle AWS risorse durante il supporto IPv6.

Che cos'è IPv6?

IPv6 è lo standard IP di nuova generazione destinato a sostituire alla fine IPv4. La versione precedente IPv4, utilizza uno schema di indirizzamento a 32 bit per supportare 4,3 miliardi di dispositivi. IPv6 utilizza invece l'indirizzamento a 128 bit per supportare circa 340 trilioni di trilioni di trilioni di dispositivi (ovvero da 2 alla 128a potenza).

Per ulteriori informazioni, consulta la IPv6pagina web VPC.

Questi sono esempi di IPv6 indirizzi:

```
2001:cdba:0000:0000:0000:0000:3257:9652 # This is a full, unabbreviated IPv6 address.
2001:cdba:0:0:0:0:3257:9652             # The same address with leading zeros in each
 group omitted
2001:cdba::3257:965                     # A compressed version of the same address.
```

Politiche IAM dual-stack (IPv4 e) IPv6

Puoi utilizzare le policy IAM per controllare l'accesso a Secrets Manager APIs e impedire che indirizzi IP esterni all'intervallo configurato accedano a Secrets Manager APIs.

Il gestore dei segreti. L'endpoint dual-stack {region} .amazonaws.com per Secrets Manager supporta entrambi e. APIs IPv6 IPv4

Se devi supportare entrambi IPv4 e, aggiorna le politiche di filtraggio degli indirizzi IPv6 IP per gestire gli indirizzi. IPv6 Altrimenti, potresti non essere in grado di connetterti a Secrets Manager tramite IPv6.

Chi dovrebbe apportare questa modifica?

Questa modifica influisce sull'utente se si utilizza il doppio indirizzamento con politiche che lo contengonoaws:sourceIp. Il doppio indirizzamento significa che la rete supporta entrambi IPv4 e IPv6.

Se utilizzi il doppio indirizzamento, aggiorna le politiche IAM che attualmente utilizzano indirizzi di IPv4 formato per includere gli indirizzi di IPv6 formato.

Chi non dovrebbe apportare questa modifica?

Questa modifica non ha effetto su di te se utilizzi solo IPv4 reti.

Aggiunta IPv6 a una policy IAM

Le policy IAM utilizzano la chiave `aws:SourceIp` condition per controllare l'accesso da indirizzi IP specifici. Se la tua rete utilizza il doppio indirizzamento (IPv4 and IPv6), aggiorna le policy IAM per includere gli intervalli di IPv6 indirizzi.

Per quanto `Condition` riguarda le policy, utilizza gli `NotIpAddress` operatori `IpAddress` and per le condizioni degli indirizzi IP. Non utilizzate operatori di stringa, poiché non sono in grado di gestire i vari formati di IPv6 indirizzi validi.

Questi esempi utilizzanoaws:SourceIp. Per VPCs, usa `aws:VpcSourceIp` invece.

Di seguito è riportata la politica di riferimento relativa all'IP di origine (Denies access to) AWS basata sulla politica di riferimento relativa all'IP di origine contenuta nella IAM User Guide. `NotIpAddress`Nell'`Condition`elemento to elenca due intervalli di IPv4 indirizzi `192.0.2.0/24` e`203.0.113.0/24`, a cui verrà negato l'accesso all'API.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Deny",
        "Action": "*",
        "Resource": "*",
        "Condition": {
            "NotIpAddress": {
                "aws:SourceIp": [
                    "192.0.2.0/24",
                    "203.0.113.0/24"
                ]
            },
            "Bool": {
                "aws:ViaAWSService": "false"
            }
        }
    }
}
```

Per aggiornare questo criterio, modifica l'`Condition`elemento in modo da includere gli intervalli di IPv6 indirizzi `2001:DB8:1234:5678::/64` e`2001:cdba:3257:8593::/64`.

> **Note**
>
> Non rimuovere gli IPv4 indirizzi esistenti. Sono necessari per la compatibilità con le versioni precedenti.

```
"Condition": {
                "NotIpAddress": {
                    "aws:SourceIp": [
                        "192.0.2.0/24", <<DO NOT REMOVE existing IPv4 address>>
                        "203.0.113.0/24", <<DO NOT REMOVE existing IPv4 address>>
                        "2001:DB8:1234:5678::/64", <<New IPv6 IP address>>
                        "2001:cdba:3257:8593::/64" <<New IPv6 IP address>>
                    ]
                },
                "Bool": {
                    "aws:ViaAWSService": "false"
                }
            }
```

Per aggiornare questa policy per un VPC, usa `aws:VpcSourceIp` invece di: `aws:SourceIp`

```
"Condition": {
                "NotIpAddress": {
                    "aws:VpcSourceIp": [
                        "10.0.2.0/24", <<DO NOT REMOVE existing IPv4 address>>
                        "10.0.113.0/24", <<DO NOT REMOVE existing IPv4 address>>
                        "fc00:DB8:1234:5678::/64", <<New IPv6 IP address>>
                        "fc00:cdba:3257:8593::/64" <<New IPv6 IP address>>
                    ]
                },
                "Bool": {
                    "aws:ViaAWSService": "false"
                }
            }
```

Verifica dei supporti del client IPv6

Se usi il gestore dei segreti. Endpoint {region} .amazonaws.com, verifica di poterti connettere ad esso. I passaggi seguenti descrivono come eseguire la verifica.

Questo esempio utilizza Linux e curl versione 8.6.0 e utilizza il Gestione dei segreti AWS servizio che ha endpoint IPv6 abilitati situati nell'endpoint amazonaws.com.

> **Note**
>
> Il gestore dei segreti. {region} .amazonaws.com si differenzia dalla tipica convenzione di denominazione dual-stack. Per un elenco completo degli endpoint di Secrets Manager, vedereGestione dei segreti AWS endpoint.
> Passa Regione AWS alla stessa regione in cui si trova il servizio. In questo esempio, utilizziamo l'endpoint Stati Uniti orientali (Virginia settentrionale) - `us-east-1`.

1. Determina se l'endpoint si risolve con un IPv6 indirizzo utilizzando il seguente comando. `dig`

```
$ dig +short AAAA secretsmanager.us-east-1.amazonaws.com

> 2600:1f18:e2f:4e05:1a8a:948e:7c08:c1c3
```

2. Determina se la rete client può stabilire una IPv6 connessione utilizzando il seguente comando. `curl` Un codice di risposta 404 indica che la connessione è riuscita, mentre un codice di risposta 0 indica che la connessione non è riuscita.

```
$ curl --ipv6 -o /dev/null --silent -w "\nremote ip: %{remote_ip}\nresponse code:
 %{response_code}\n" https://secretsmanager.us-east-1.amazonaws.com

> remote ip: 2600:1f18:e2f:4e05:1a8a:948e:7c08:c1c3
> response code: 404
```

Se è stato identificato un IP remoto e il codice di risposta no`0`, è stata stabilita correttamente una connessione di rete all'endpoint utilizzando. IPv6 L'IP remoto deve essere un IPv6 indirizzo perché il sistema operativo deve selezionare il protocollo valido per il client.

Se l'IP remoto è vuoto o il codice di risposta lo è`0`, la rete client o il percorso di rete verso l'endpoint è IPv4 -only. È possibile verificare questa configurazione con il seguente `curl` comando.

```
$ curl -o /dev/null --silent -w "\nremote ip: %{remote_ip}\nresponse code:
 %{response_code}\n" https://secretsmanager.us-east-1.amazonaws.com

> remote ip: 3.123.154.250
> response code: 404
```

Resilienza in Gestione dei segreti AWS

AWS costruisce l'infrastruttura globale attorno alle zone Regioni AWS di disponibilità. Regioni AWS forniscono più zone di disponibilità fisicamente separate e isolate, che si collegano a reti a bassa latenza, ad alto throughput e altamente ridondanti. Con le zone di disponibilità è possibile progettare e gestire applicazioni e database che eseguono automaticamente il failover tra zone di disponibilità senza interruzioni. Le zone di disponibilità consentono una maggiore disponibilità, tolleranza ai guasti e scalabilità rispetto alle infrastrutture a data center singolo o multiplo.

Per ulteriori informazioni sulla resilienza e il disaster recovery, consulta Reliability Pillar — Well-Architected AWS Framework.

Per ulteriori informazioni sulle zone di disponibilità, Regioni AWS vedere Global Infrastructure.AWS

TLS post-quantistico

Secrets Manager supporta un'opzione di scambio di chiavi post-quantistiche ibride per il protocollo di crittografia di rete Transport Layer Security (TLS). Puoi utilizzare questa opzione TLS quando ti connetti agli endpoint API di Secrets Manager. Offriamo questa funzionalità prima che gli algoritmi post-quantistici siano standardizzati in modo da poter iniziare a testare l'effetto di questi protocolli di scambio di chiavi sulle chiamate di Secrets Manager. Queste caratteristiche opzionali di scambio di chiavi post-quantistiche ibride sono sicure almeno quanto la crittografia TLS che utilizziamo oggi e potrebbero fornire ulteriori vantaggi per la sicurezza. Tuttavia, influenzano la latenza e il throughput rispetto ai protocolli di scambio di chiavi classici in uso oggi. Per impostazione predefinita, l'agente Secrets Manager utilizza lo scambio di chiavi ML-KEM post-quantistico come scambio di chiavi con la massima priorità.

Per proteggere i dati crittografati oggi da potenziali attacchi futuri, AWS partecipa con la comunità crittografica allo sviluppo di algoritmi quantistici resistenti o post-quantistici. Abbiamo implementato suite di crittografia di scambio di chiavi post-quantistiche ibride negli endpoint di Secrets Manager. Queste suite di crittografia ibride, che combinano elementi classici e post-quantistici, assicurano

che la connessione TLS sia potente almeno quanto lo sarebbe con le suite di crittografia classiche. Tuttavia, poiché le funzionalità delle prestazioni e i requisiti di larghezza di banda delle suite di crittografia ibride sono diversi da quelli dei classici meccanismi di scambio di chiavi, consigliamo di testarli nelle chiamate API.

Secrets Manager supporta PQTLS in tutte le Regioni, ad eccezione di quelle cinesi.

Per configurare il protocollo TLS post-quantistico ibrido

1. Aggiungi il client AWS Common Runtime alle tue dipendenze Maven. Si consiglia di utilizzare l'ultima versione disponibile. Ad esempio, questa istruzione aggiunge la versione 2.20.0.

   ```
   <dependency>
     <groupId>software.amazon.awssdk</groupId>
     <artifactId>aws-crt-client</artifactId>
     <version>2.20.0</version>
   </dependency>
   ```

2. Aggiungi l' AWS SDK for Java 2.x al tuo progetto e inizializzalo. Abilita le suite di crittografia post-quantistica ibrida su tuo client HTTP.

   ```
   SdkAsyncHttpClient awsCrtHttpClient = AwsCrtAsyncHttpClient.builder()
               .postQuantumTlsEnabled(true)
               .build();
   ```

3. Crea il client asincrono di Secrets Manager.

   ```
   SecretsManagerAsyncClient SecretsManagerAsync = SecretsManagerAsyncClient.builder()
               .httpClient(awsCrtHttpClient)
               .build();
   ```

   Ora quando richiami le operazioni API di Secrets Manager, le chiamate vengono trasmesse all'endpoint di Secrets Manager utilizzando TLS post-quantistico ibrido.

Per ulteriori informazioni sull'utilizzo del protocollo TLS post-quantistico ibrido, consulta:

- AWS SDK for Java 2.x Guida per gli sviluppatori e post AWS SDK for Java 2.x pubblicato sul blog.
- Presentazione di s2n-tls, una nuova implementazione TLS open source e Utilizzo di s2n-tls.
- Crittografia post-quantistica sul National Institute for Standards and Technology (NIST).

- Post-Quantum Key Encapsulation Methods (PQ KEM) ibrido per Transport Layer Security (TLS) 1.2.

Il TLS post-quantum per Secrets Manager è disponibile in tutti i paesi tranne Regioni AWS la Cina.

Risoluzione dei problemi Gestione dei segreti AWS

Utilizza le informazioni contenute in questa pagina per diagnosticare e risolvere i problemi che possono verificarsi durante l'utilizzo di ruoli con Secrets Manager.

Per i problemi relativi alla rotazione, consulta the section called "Risoluzione dei problemi della rotazione".

Argomenti

Messaggi di «Accesso negato»

Quando effettui una chiamata API come GetSecretValue o CreateSecret verso Secrets Manager, devi disporre delle autorizzazioni IAM per effettuare quella chiamata. Quando usi la console, la console effettua le stesse chiamate API per tuo conto, quindi devi disporre anche delle autorizzazioni IAM. Un amministratore può concedere le autorizzazioni allegando una policy IAM al tuo utente IAM o a un gruppo di cui sei membro. Se le dichiarazioni politiche che concedono tali autorizzazioni includono condizioni, ad time-of-day esempio restrizioni relative all'indirizzo IP, devi soddisfare anche tali requisiti al momento dell'invio della richiesta. Per informazioni sulla visualizzazione o sulla modifica delle policy per un ruolo, un gruppo o un utente IAM, consulta Lavorare con le policy nella Guida per l'utente di IAM. Per informazioni sulle autorizzazioni richieste per Secrets Manager, consulta the section called "Autenticazione e controllo degli accessi".

Se stai firmando le richieste API manualmente, senza utilizzare il AWS SDKs, verifica di aver firmato correttamente la richiesta.

"Accesso negato" per le credenziali di sicurezza temporanee

Verifica che l'utente o il ruolo IAM utilizzato per effettuare la richiesta disponga delle autorizzazioni corrette. Autorizzazioni per credenziali di sicurezza temporanee derivano da un utente o ruolo IAM. Questo significa che le autorizzazioni sono limitate a quelle concesse al ruolo o all'utente IAM. Per ulteriori informazioni su come sono determinate le autorizzazioni per le credenziali di sicurezza provvisorie, consulta controllo delle autorizzazioni per le credenziali di sicurezza provvisorie nella Guida IAM per lo sviluppatore.

Verifica che le richieste vengano firmate correttamente e che il formato della richiesta sia valido. Per i dettagli, consulta la documentazione del toolkit per l'SDK scelto o Using Temporary Security Credentials to Request Access to AWS Resources nella IAM User Guide.

Verifica che le credenziali di sicurezza provvisorie non siano scadute. Per ulteriori informazioni, consulta Richiesta di credenziali di sicurezza provvisorie nella Guida per l'utente di IAM.

Per informazioni sulle autorizzazioni richieste per Secrets Manager, consulta the section called "Autenticazione e controllo degli accessi".

Le modifiche apportate non sono sempre immediatamente visibili.

Secrets Manager utilizza un modello di calcolo distribuito chiamato consistenza finale. Qualsiasi modifica apportata in Secrets Manager (o in altri AWS servizi) richiede tempo per diventare visibile da tutti gli endpoint possibili. Alcuni dei ritardi sono dovuti al tempo necessario per inviare i dati da un server a un altro, da una zona di replica a un'altra e da una regione a un'altra nel mondo. Secrets Manager utilizza inoltre la memorizzazione nella cache per migliorare le prestazioni, è possibile che ciò aumenti il tempo. in quanto la modifica potrebbe risultare visibile solo dopo il timeout dei dati memorizzati nella cache.

Progetta le tue applicazioni globali in modo da considerare questi potenziali ritardi e assicurati che funzionino come previsto, anche quando una modifica apportata in una posizione non è immediatamente visibile in un'altra.

Per ulteriori informazioni su come alcuni altri AWS servizi sono influenzati dall'eventuale coerenza, consulta:

- Gestione della consistenza dei dati nella Guida per sviluppatori di Amazon Redshift Database
- Modello di consistenza dati di Amazon S3 nella Guida per l'utente di Amazon Simple Storage Service

- Garantire la consistenza quando si utilizzano Amazon S3 e Amazon EMR per flussi di lavoro ETL nel blog dei Big Data AWS .
- Consistenza finale Amazon EC2 nella Riferimento all'API di Amazon EC2

"Impossibile generare una chiave dati con una chiave KMS asimmetrica" durante la creazione di un segreto

Secrets Manager utilizza una chiave KMS di crittografia simmetrica associata a un segreto per generare una chiave di dati per ogni valore del segreto. Non puoi utilizzare una chiave KMS asimmetrica. Verifica di utilizzare una chiave KMS di crittografia simmetrica anziché una chiave KMS asimmetrica. Per le istruzioni, consulta Individuazione delle chiavi KMS asimmetriche.

Un'operazione AWS CLI o AWS SDK non riesce a trovare il mio segreto da un ARN parziale

In molti casi, Secrets Manager può trovare il segreto da una parte di un ARN anziché dall'ARN completo. Tuttavia, se il nome del tuo segreto termina con un trattino seguito da sei caratteri, Secrets Manager potrebbe non essere in grado di individuare il segreto da una sola parte di un ARN. Invece, ti consigliamo di utilizzare l'ARN completo o il nome del segreto.

Ulteriori dettagli

Secrets Manager include sei caratteri casuali alla fine del nome del segreto per garantire che l'ARN del segreto sia univoco. Se il segreto originale viene eliminato e quindi viene creato un nuovo segreto con lo stesso nome, i due segreti sono diversi a ARNs causa di questi caratteri. Gli utenti con accesso al vecchio segreto non ottengono automaticamente l'accesso al nuovo segreto perché ARNs sono diversi.

Secrets Manager costruisce un ARN per un segreto con Regione, account, nome segreto e poi un trattino e altri sei caratteri, come segue:

```
arn:aws:secretsmanager:us-east-2:111122223333:secret:SecretName-abcdef
```

Se il tuo nome segreto termina con un trattino e sei caratteri, l'utilizzo di una sola parte dell'ARN può apparire in Secrets Manager come se si stesse specificando un ARN completo. Ad esempio, potresti avere un segreto denominato `MySecret-abcdef` con l'ARN

`arn:aws:secretsmanager:us-east-2:111122223333:secret:MySecret-abcdef-nutBrk`

Se si chiama la seguente operazione, che utilizza solo una parte dell'ARN segreto, Secrets Manager potrebbe non trovare il segreto.

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

Questo segreto è gestito da un AWS servizio ed è necessario utilizzare tale servizio per aggiornarlo.

Se questo messaggio viene visualizzato mentre provi a modificare un segreto, il segreto potrà essere aggiornato solo utilizzando il servizio di gestione riportato nel messaggio. Per ulteriori informazioni, consulta Segreti gestiti da altri servizi.

Per determinare chi gestisce un segreto, puoi rivedere il nome del segreto. I segreti gestiti da altri servizi sono preceduti dall'ID di quel servizio. Oppure, chiama describe-secret AWS CLI, quindi esamina il campo. `OwningService`

L'importazione del modulo Python fallisce quando si utilizza **`Transform: AWS::SecretsManager-2024-09-16`**

Se stai usando Transform: `AWS::SecretsManager-2024-09-16` e riscontri errori di importazione del modulo Python durante l'esecuzione della funzione Lambda di rotazione, il problema è probabilmente causato da un valore incompatibile. `Runtime` Con questa versione di trasformazione, AWS CloudFormation gestisce automaticamente la versione di runtime, il codice e i file di oggetti condivisi. Non è necessario gestirli da soli.

Gestione dei segreti AWS quote

Secrets Manager read APIs ha quote TPS elevate e i piani di controllo APIs che vengono chiamati meno frequentemente hanno quote TPS inferiori. Ti consigliamo di evitare di chiamare `PutSecretValue` o `UpdateSecret` ad una frequenza sostenuta di più di una volta ogni 10 minuti. Quando chiami `PutSecretValue` o `UpdateSecret` per aggiornare il valore del segreto, Secrets Manager crea una nuova versione del segreto. Secrets Manager rimuove le versioni obsolete quando sono più di 100, ma non rimuove le versioni create da meno di 24 ore. Se aggiorni il valore segreto più di una volta ogni 10 minuti, crei più versioni di quelle che Secrets Manager rimuove e raggiungerai la quota massima prevista per le versioni di un segreto.

Puoi utilizzare più regioni nel tuo account e ogni quota sarà specifica per ogni regione.

Quando un'applicazione in un'unica applicazione Account AWS utilizza un segreto di proprietà di un altro account, si parla di richiesta tra account. Per le richieste tra account, Secrets Manager limita l'account che effettua le richieste di identità, non l'account proprietario del segreto. Ad esempio, se un'identità nell'account A utilizza un segreto nell'account B, l'uso del segreto viene applicato solo alle quote dell'account A.

Quote di Secrets Manager

Name	Predefinita	Adatta e	Description
Frequenza combinata di DeleteResourcePolicy GetResourcePolicy, PutResourcePolicy, e richieste ValidateResourcePolicy API	Ogni regione supportata: 50 al secondo	No	Il numero massimo di transazioni al secondo per DeleteResourcePolicy GetResourcePolicy PutResourcePolicy, e le richieste ValidateResourcePolicy API combinate.
Frequenza combinata di richieste PutSecretValue RemoveRegionsFromReplication ReplicateSecretToRegion	Ogni regione supportata: 50 al secondo	No	Il numero massimo di transazioni al secondo per PutSecretValue RemoveRegionsFromR

Name	Predefinita	Adatta e	Description
StopReplicationToReplica, UpdateSecret,, e UpdateSecretVersionStage API			eplication, ReplicateSecretToRegion, StopReplicationToReplica UpdateSecret, e le richieste UpdateSecretVersionStage API combinate.
Frequenza combinata di richieste RestoreSecret API	Ogni regione supportata: 50 al secondo	No	Il numero massimo di transazioni al secondo per le richieste RestoreSecret API.
Frequenza combinata di richieste CancelRotateSecret API RotateSecret e velocità	Ogni regione supportata: 50 al secondo	No	Il numero massimo di transazioni al secondo RotateSecret e le richieste CancelRotateSecret API combinate.
Frequenza combinata delle richieste UntagResource API TagResource e	Ogni regione supportata: 50 al secondo	No	Il numero massimo di transazioni al secondo TagResource e le richieste UntagResource API combinate.
Frequenza delle richieste BatchGetSecretValue API	Ogni regione supportata: 100 al secondo	No	Il numero massimo di transazioni al secondo per le richieste BatchGetSecretValue API.
Frequenza delle richieste CreateSecret API	Ogni regione supportata: 50 al secondo	No	Il numero massimo di transazioni al secondo per le richieste CreateSecret API.

Name	Predefinita	Adatta e	Description
Frequenza delle richieste DeleteSecret API	Ogni regione supportata: 50 al secondo	No	Il numero massimo di transazioni al secondo per le richieste DeleteSecret API.
Frequenza delle richieste DescribeSecret API	Ogni regione supportata: 40.000 al secondo	No	Il numero massimo di transazioni al secondo per le richieste DescribeSecret API.
Frequenza delle richieste GetRandomPassword API	Ogni regione supportata: 50 al secondo	No	Il numero massimo di transazioni al secondo per le richieste GetRandomPassword API.
Frequenza delle richieste GetSecretValue API	Ogni regione supportata: 10.000 al secondo	No	Il numero massimo di transazioni al secondo per le richieste GetSecretValue API.
Frequenza delle richieste ListSecretVersionIds API	Ogni regione supportata: 50 al secondo	No	Il numero massimo di transazioni al secondo per le richieste ListSecretVersionIds API.
Frequenza delle richieste ListSecrets API	Ogni regione supportata: 100 al secondo	No	Il numero massimo di transazioni al secondo per le richieste ListSecrets API.

Name	Predefinita	Adatta e	Description
Lunghezza policy basata su risorse	Ogni regione supportata: 20.480	No	Il numero massimo di caratteri in una policy di autorizzazioni basata su risorse collegata a un segreto.
Dimensione del valore del segreto	Ogni regione supportata: 65.536 byte	No	La dimensione massima di un valore del segreto crittografato. Se il valore del segreto è una stringa, questo è il numero di caratteri consentiti nel valore del segreto.
Segreti	Ogni regione supportata: 500.000	No	Il numero massimo di segreti in ogni AWS regione di questo AWS account.
Etichette allegate su tutte le versioni di un segreto	Ogni regione supportata: 20	No	Il numero massimo di etichette di staging allegate su tutte le versioni di un segreto.
Versioni per segreto	Ogni regione supportata: 100	No	Il numero massimo di versioni di un segreto.

Aggiungi tentativi alla tua applicazione

Il tuo AWS cliente potrebbe vedere fallire le chiamate a Secrets Manager a causa di problemi imprevisti sul lato client. Il tuo potrebbe vedere che le chiamate a Secrets Manager non riescono a causa della limitazione della velocità. Quando superi una quota di richiesta API, Secrets Manager limita la richiesta. Rifiuta una richiesta altrimenti valida e restituisce unthrottlingErrore. Per entrambi

i tipi di guasti, ti consigliamo di riprovare la chiamata dopo un breve periodo di attesa. Questo è chiamato astrategia di backoff e riprova.

Se si verificano errori come quelli riportati di seguito, potrebbe essere necessario aggiungere al codice dell'applicazione:

Errori ed eccezioni transitori

- `RequestTimeout`
- `RequestTimeoutException`
- `PriorRequestNotComplete`
- `ConnectionError`
- `HTTPClientError`

Limitazione lato servizio e limitazione di errori ed eccezioni

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

Per ulteriori informazioni, oltre al codice di esempio, su tentativi, backoff esponenziale e jitter, vedere le seguenti risorse:

- Backoff esponenziale e jitter
- Timeout, tentativi e backoff con jitter

- Tentativi di errore e backoff esponenziale in entrata. AWS

Cronologia dei documenti

La tabella seguente descrive le modifiche importanti alla documentazione dall'ultima versione di Gestione dei segreti AWS. Per ricevere notifiche sugli aggiornamenti di questa documentazione, è possibile iscriversi a un feed RSS.

Modifica	Descrizione	Data
Nuova politica AWS gestita	Secrets Manager ha rilasciato una nuova policy gestita `AWSSecretsManagerClientReadOnlyAccess` che fornisce l'accesso in sola lettura ai segreti per le applicazioni client. Per informazioni, consulta gli aggiornamenti di Secrets Manager alle policy AWS gestite.	5 novembre 2025
È stato aggiunto il supporto per i tag di allocazione dei costi	Secrets Manager ora supporta i tag di allocazione dei costi, che consentono ai clienti di classificare e tenere traccia dei costi per reparto, team o applicazione. Per ulteriori informazioni, vedere Utilizzo dei tag di allocazione dei costi con. Gestione dei segreti AWS	27 maggio 2025
Supporto aggiuntivo IPv6 e dual-stack	Secrets Manager ora supporta gli endpoint dual-stack. Visualizza IPv4 e IPv6 accedi per ulteriori informazioni.	20 dicembre 2024
Passaggio da Secrets Manager a policy AWS gestita	La policy `SecretsManagerReadWrite` gestita	12 marzo 2024

	ora include `redshift-serverless` l'autorizzazione. Per ulteriori informazioni, consulta la politica AWS gestita per Gestione dei segreti AWS	

Aggiornamenti precedenti

La tabella seguente descrive le modifiche importanti apportate in ogni versione della Guida per l' Gestione dei segreti AWS utente prima di febbraio 2024.

Modifica	Descrizione	Data
Disponibilità generale	Questa è la versione pubblica iniziale di Secrets Manager.	4 aprile 2018

Le traduzioni sono generate tramite traduzione automatica. In caso di conflitto tra il contenuto di una traduzione e la versione originale in Inglese, quest'ultima prevarrà.